



DB2 版本 9

适用于 Linux、UNIX 和 Windows



系统监视器指南和参考

s151-0283-00





DB2 版本 9

适用于 Linux、UNIX 和 Windows



系统监视器指南和参考

s151-0283-00

在使用本资料及其支持的产品之前，请务必阅读『*声明*』中的一般信息。

版本声明

本文档包含 IBM 的专利信息。它是根据许可协议提供的，并受版权法保护。本出版物包含的信息不包括任何产品保证，且本手册提供的任何声明不应作如此解释。

可以用在线方式或通过您当地的 IBM 代表订购 IBM 出版物。

- 要以在线方式订购出版物，可访问 IBM 出版物中心（IBM Publications Center），网址为 www.ibm.com/shop/publications/order。
- 要查找您当地的 IBM 代表，可访问 IBM 全球联系人目录（IBM Directory of Worldwide Contacts），网址为 www.ibm.com/planetwide。

在美国或加拿大，要从“DB2 市场营销和销售中心”订购 DB2 出版物，请致电 1-800-IBM-4YOU（426-4968）。

当您发送信息给 IBM 后，即授予 IBM 非专有权，IBM 对于您所提供的任何信息，有权利以任何它认为适当的方式使用或分发，而不必对您负任何责任。

目录

第 1 部分 系统监视器指南

第 1 章 数据库系统监视器简介

数据库系统监视器

对于数据库管理系统的性能和运行状况的维护而言，数据库监视是一个非常重要的活动。为便于进行监视，DB2® 从数据库管理器、数据库及所有已连接的应用程序收集信息。可借助此信息执行下列及其他任务:

- 根据数据库使用模式预计硬件需求。
- 分析各个应用程序或 SQL 查询的性能。
- 跟踪索引和表的使用情况。
- 查明系统性能下降的原因。
- 评估优化活动（如改变数据库管理器配置参数、添加索引或修改 SQL 查询）的效果。

主要有两个工具可用来访问系统监视器信息: 快照监视器和事件监视器; 每个工具都有不同的用途。快照监视器使您能够在特定点及时（生成快照时）捕获数据库活动的状态图片。事件监视器会在发生指定数据库事件时记录数据。

系统监视器提供了多种方法来显示监视器数据。对于快照监视器和事件监视器，您可以选择将监视器信息存储在文件或 SQL 表中，在屏幕上显示这些信息（将其直接导至标准输出），或者使用客户机应用程序来处理它们。

相关概念:

- 第 6 页的『计数器状态和可视性』
- 第 3 页的『数据库系统监视器数据结构』
- 第 8 页的『数据库系统监视器内存需求』
- 第 55 页的『事件监视器』
- 第 19 页的『快照监视器』

数据库系统监视器数据结构

数据库系统监视器将收集的信息存储在称为*监视元素*（先前称为数据元素）的实体中。每个监视元素存储有关数据库系统状态的一个特定方面的信息。此外，监视元素由唯一名称标识并且存储特定类型的信息。

以下是监视元素用来存储数据的可用元素类型:

计数器
: 计算活动的发生次数。在监视期间，计数器值会增大。大多数计数器元素可以复位。

标尺
: 指示某个项的当前值。根据数据库活动的不同，标尺值会增加或减小（例如，挂起的锁定数）。标尺元素不能复位。

水位标记
: 指示自开始监视以来元素所达到的最高值（最大值）或最低值（最小值）。水位标记元素不能复位。

信息
: 提供参考类型的监视活动详细信息。这可以包括诸如分区名称、别名和路径详细信息之类的项。信息元素不能复位。

时间戳记
: 通过提供自 1970 年 1 月 1 日后经历的秒数和微秒数，以指示活动发生的日期和时间。对于快照监视器和事件监视器，时间戳记元素的收集由 TIMESTAMP 监视开关控制。此开关在缺省情况下设置为 ON，如果数据库实例的 CPU 利用率达到 100%，则考虑性能方面的原因应将其设置为 OFF。时间戳记元素不能复位。

  时间戳记元素的值为零意味着“不可用”。如果尝试导入此数据，这样的值将生成超过范围错误（SQL0181）。为避免此错误，在导出数据前将该值更新为有效时间戳记值。

时间
: 返回执行活动时耗用的秒数和微秒数。对于快照监视器和事件监视器，大多数时间元素的收集由 TIMESTAMP 监视开关控制。此开关在缺省情况下设置为 ON，如果数据库实例的 CPU 利用率达到 100%，则考虑性能方面的原因应将其设置为 OFF。某些时间元素可以复位。

监视元素收集一个或多个逻辑数据组的数据。逻辑数据组是一组监视元素，它们用来收集特定数据库活动作用域的数据库系统监视信息。监视元素在逻辑数据组中按它们提供的信息级别排序。例如，在进行快照监视时，“总排序时间”监视元素将返回数据库（dbase）、应用程序（appl）和语句（stmt）信息；因此，它将出现在已列示并且用圆括号括起来的每个逻辑数据组中。

尽管许多监视元素同时被快照监视器和事件监视器使用，但每个监视器使用一个不同的逻辑数据组集合。这是因为可对其捕获快照的数据库活动作用域与可对其收集事件数据的数据库活动作用域不同。从实际上说，可从快照监视器访问的完整监视元素集合不同于可从事件监视器访问的那些监视元素。

相关概念:

- 第 6 页的『计数器状态和可视性』
- 第 3 页的『数据库系统监视器』
- 第 55 页的『事件监视器』
- 第 19 页的『快照监视器』
- 第 11 页的『系统监视开关』

相关参考:

- 『RESET MONITOR command』（*Command Reference*）

比较 DB2 监视器

DB2 版本 9.1 提供了可用来监视数据库系统的多种方法。快照监视器、事件监视器和运行状况监视器中的每一个都能满足不同监视需要。下表提供不同监视器的简短概述并比较它们的特征。

表 1. 比较 DB2 版本 9.1 监视器

	快照监视器	事件监视器	运行状况监视器
描述	• 实时监视。 • 提供当前时间点的数据库状态的视图。 • 返回的数据可用来检查数据库状态和标识潜在问题区域。以一定时间间隔进行捕获时，可以显示数据库活动的趋势。	• 基于触发器的实时监视。 • 每次发生特定类型的事件时记录数据库的状态，描述一段时间内的数据库活动。 • 提供用于查明／诊断问题的详细数据	• 基于异常情况的监视。 • 标志数据库中异常中止或存在潜在问题的情况。 • 提供高级数据库运行状况视图。指出将来进行调查时需要注意的常规区域。
所收集数据的级别	• 数据库管理器 • 数据库 • 应用程序（包括语句级别信息） • 缓冲池 • 表空间 • 表 • 锁定和锁定等待 • 动态 SQL • DCS 应用程序和数据库	• 数据库 • 连接（相当于快照应用程序级别） • 缓冲池 • 表空间 • 表 • 死锁 • 事务 • 语句	• 数据库管理器 • 数据库 • 表空间 • 表空间容器
Activated/enabled	在缺省情况下将特定监视开关设置为 ON [1]；TIMESTAMP=ON。可对每个应用程序（使用 **update monitor switches** 命令）或在数据库管理器级别（使用 **update dbm cfg** 命令）启用开关。	使用 AUTOSTART 选项创建事件监视器，或者将事件监视器设置为状态 1[2]。	在缺省情况下启用。要释放，请将 *health_mon* 数据库管理器配置参数设置为 OFF
收集数据的时间	用户发出快照表函数、快照 API、SNAP_WRITE_FILE 存储过程，或者从 CLP 获取快照命令，或者从快照管理视图发出 SELECT	指定的事件发生次数 [3]	缺省情况下按预设时间间隔进行收集

表 1. 比较 DB2 版本 9.1 监视器 (续)

	快照监视器	事件监视器	运行状况监视器
检索／分析数据的手段	使用快照表函数、快照管理视图、CLP、快照监视器 API 或活动监控器图形工具。	• 对于表事件监视器，使用 SQL 访问事件表或使用事件分析器图形工具 • 对于命名管道事件监视器，使用 db2evmon 实用程序或从管道读取监视器数据的客户机程序 • 对于文件事件监视器，使用事件分析器、db2evmon 实用程序或从文件读取监视器数据的客户机程序	• 接收有关警报的电子邮件或寻呼机通知 • 使用 SQL 表函数、CLP 快照和运行状况快照 API 来检索运行状况数据 • 使用“运行状况中心”来查看当前警报 • 通过使用建议顾问程序图形工具解决警报，并从 CLP、存储过程或 API 返回建议
开销	随启用的开关数和实例上运行的工作负载类型的不同而变化；可能会增加 3-10% 的系统工作负载	随要监视的数据类型（如语句事件监视器对每个执行的语句返回详细数据）和事件监视器的选择方式（如是否使用 WHERE 子句）的不同而变化	运行状况监视的最低开销。从“运行状况中心”调用的图形工具导致的附加开销。

注:

1. 有些监视器信息不受开关控制，而是一直进行收集。仅当特定开关设置为 ON 时，才会收集其他类型的监视器信息。
2. 在缺省情况下，将对每个数据库创建详细信息死锁事件监视器 DB2DETAILDEADLOCK，并在数据库激活时启动该监视器。
3. 可清空事件监视器缓冲区以强制事件监视器写出其当前数据。

计数器状态和可视性

数据库管理器收集的监视元素包括若干累加计数器。这些计数器将在数据库或数据库管理器操作期间递增，如每次应用程序落实事务时。

将在适用对象可用时初始化计数器。例如，对数据库读取的缓冲池页数（基本监视元素）在数据库激活时设置为零。

某些计数器由监视开关控制。如果特定监视开关设置为“OFF”，则受其控制的监视元素不会收集数据。当监视开关设置为“ON”时，所有关联计数器都将复位为零。

当事件监视器激活时，事件监视器返回的计数器将复位为零。

事件监视器计数表示自下列其中一个起始点之后的计数:
- 对数据库、表空间和表启动事件监视器时。

- 对现有连接启动事件监视器时。
- 应用程序连接，对于在启动监视器之后建立的连接。
- 在启动监视器之后又启动下一个事务（工作单元）或语句时。
- 在启动监视器之后发生死锁时。

每个事件监视器和任意监视应用程序（使用快照监视器 API 的应用程序）有自己的系统监视器数据逻辑视图。这意味着复位或初始化计数器时，将仅影响复位或初始化计数器的事件监视器或应用程序。除非关闭事件监视器然后再打开它，否则事件监视器计数器不能复位。获取快照的应用程序可使用 RESET MONITOR 命令随时复位其计数器视图。

如果在语句启动之后启动语句事件监视器，则监视器将在启动下一条 SQL 语句时开始收集信息。因此，事件监视器不会返回有关启动监视器时数据库管理器正在执行的语句的信息。对于事务信息也如此。

相关概念:

- 第 11 页的『系统监视开关』
- 第 3 页的『数据库系统监视器』
- 第 3 页的『数据库系统监视器数据结构』
- 第 55 页的『事件监视器』
- 第 19 页的『快照监视器』

相关参考:

- 『RESET MONITOR command』（*Command Reference*）

系统监视器输出：自描述数据流

除了在屏幕上显示监视器数据或将其存储在 SQL 表中之外，还可以开发客户机应用程序来处理它。系统监视器通过自描述数据流同时对快照监视器和事件监视器返回监视器数据。在快照监视应用程序中，可调用快照 API 以捕获快照，然后直接处理数据流。

处理事件监视器数据的不同之处在于发生数据库事件时事件数据将发送至应用程序。对于管道事件监视器，应用程序将等待事件数据到达，并在事件数据到达时处理它。对于文件事件监视器，应用程序将对事件文件进行语法分析，因此会分批处理事件记录。

此自描述数据流允许您对返回的数据进行语法分析，一次分析一个元素。这样就可以进行多方面的监视，包括查找有关特定应用程序或特定数据库状态的信息。

返回的监视器数据具有以下格式:

大小
存储在监视元素或逻辑数据分组中的数据的大小（以字节计）。如果是逻辑数据分组，则此项为逻辑组中的所有数据的大小。例如，数据库逻辑分组（*db*）包含各个监视元素（如 *total_log_used*）及其他逻辑数据分组，如前滚信息信息（*rollforward*）。这不包括“大小”、“类型”和“元素”信息占用的大小。

类型
存储在数据中的元素的类型（如可变长度字符串或带符号的 32 位数字值）。元素类型头指的是元素的逻辑数据分组。

元素标识
监视器捕获的监视元素的标识。如果是逻辑数据分组，则此项为该组的标识（如 *collected*、*dbase* 或 *event_db*）。

数据
监视器对监视元素收集的值。如果是逻辑数据分组，则该数据由属于它的监视元素组成。

监视元素中的所有时间戳记将以两个不带符号的 4 字节监视元素（秒和微秒）的形式返回。它们表示 GMT 时间 1970 年 1 月 1 日。

监视元素中的字符串的大小元素表示该字符串元素的实际数据大小。因此字符串不是以 NULL 终止的，所以此大小不包括 NULL 终止符。

相关概念:

- 第 85 页的『事件监视器自描述数据流』
- 第 124 页的『事件类型至逻辑数据组的映射』
- 第 17 页的『监视开关自描述数据流』
- 第 49 页的『快照监视器自描述数据流』

相关参考:

- 第 93 页的『快照监视器接口至逻辑数据组的映射』

数据库系统监视器内存需求

将从监视器堆分配维护数据库系统监视器数据所需的内存。监视器堆大小由 `mon_heap_sz` 配置参数控制。因为下列因素，监视活动所需的内存量变化很大:

- 监视应用程序的数目
- 事件监视器的数目和特征
- 设置的监视开关
- 数据库活动级别

如果监视器命令失败并且 SQLCODE 为 -973，则考虑增加 `mon_heap_sz` 的值。

以下公式提供监视器堆所需的大致页数:

```
(应用程序使用的存储空间            +
 事件监视器使用的存储空间          +
 监视应用程序使用的存储空间        +
 网关应用程序使用的存储空间)      / 4096
```

每个应用程序使用的存储空间:

- 如果 STATEMENT 开关设置为 OFF 或零
- 如果 STATEMENT 开关设置为 ON:
  – 为将要同时运行的每个语句添加 400 字节（即，应用程序可能具有的打开游标数）。这不是应用程序已经运行的累积语句总数。
  – 如果是分区数据库，则为每个语句添加以下大小:
    - 200 字节 * (平均子节数)
- 如果应用程序发出 sqleseti() info，则添加用户标识、应用程序名称、工作站名称和记帐字符串大小。

每个事件监视器使用的存储空间

- 1300 字节
- 2 * BUFFERSIZE
- 如果事件监视器写至文件，则添加 308 字节。
- 如果事件监视器的类型为 DATABASE，则：
  - 添加 2700 字节
  - 对语句高速缓存中的每个语句添加 100 字节
- 如果事件监视器的类型为 TABLES，则：
  - 添加 600 字节
  - 对访问的每个表添加 75 字节
- 如果事件监视器的类型为 TABLESPACES，则：
  - 添加 10 字节
  - 对每个表空间添加 250 字节
- 如果事件监视器的类型为 BUFFERPOOLS，则：
  - 添加 10 字节
  - 对每个缓冲池添加 250 字节
- 如果事件监视器的类型为 CONNECTIONS：
  - 添加 600 字节
  - 对于每个已连接应用程序：
    - 添加 600 字节
  - 记住添加上述“应用程序使用的存储空间”
- 如果事件监视器的类型为 DEADLOCK：
  - 并且 WITH DETAILS HISTORY 正在运行：
    - 添加 X*100 字节，添加次数为希望运行的当前应用程序的最大数目，其中 X 是应用程序的工作单元中的最大语句数。
  - 并且 WITH DETAILS HISTORY VALUES 正在运行：
    - 同时添加 X*Y 字节，添加次数为希望运行的当前应用程序的最大数目，其中 Y 是将绑定至 SQL 语句的参数值的最大期望大小

每个监视应用程序使用的存储空间:

- 250 字节
- 对于要复位的每个数据库：
  - 350 字节
  - 对每个远程数据库添加 200 字节。
  - 如果 SORT 开关设置为 ON，则添加 25 字节。
  - 如果 LOCK 开关设置为 ON，则添加 25 字节。
  - 如果 TABLE 开关设置为 ON：
    - 添加 600 字节
    - 对访问的每个表添加 75 字节
  - 如果 BUFFERPOOL 开关设置为 ON：
    - 添加 300 字节
    - 对访问的每个表空间添加 250 字节

- 对访问的每个缓冲池添加 250 字节
- 如果 STATEMENT 开关设置为 ON:
  - 添加 2100 字节
  - 对每个语句添加 100 字节
- 对连接至数据库的每个应用程序:
  - 添加 600 字节
  - 对应用程序连接至的每个远程数据库添加 200 字节
  - 如果 SORT 开关设置为 ON，则添加 25 字节
  - 如果 LOCK 开关设置为 ON，则添加 25 字节
  - 如果 BUFFERPOOL 开关设置为 ON，则添加 250 字节

- 对于要复位的每个 DCS 数据库:
  - 对数据库添加 200 字节
  - 对连接至数据库的每个应用程序添加 200 字节
  - 如果 STATEMENT 开关设置为 ON，则必须复位传输级别数据:
    - 对于每个数据库，对每个传输级别添加 200 字节
    - 对于每个应用程序，对每个传输级别添加 200 字节

网关应用程序使用的存储空间:

- 对每个主机数据库添加 250 字节（即使所有开关设置为 OFF）
- 对每个应用程序添加 400 字节（即使所有开关设置为 OFF）
- 如果 STATEMENT 开关设置为 ON:
  - 对于每个应用程序，对同时运行的每个语句添加 200 字节（即应用程序可能具有的打开游标数）。这不是应用程序已经运行的累积语句总数。
  - 必须计算传输级别数据:
    - 对于每个数据库，对每个传输级别添加 200 字节
    - 对于每个应用程序，对每个传输级别添加 200 字节
- 如果 UOW 开关设置为 ON:
  - 则对每个应用程序添加 50 字节
- 对于使用 TMDB 的每个应用程序（用于 SYNCPOINT TWOPHASE 活动）:
  - 添加 20 字节并加上 XID 本身的大小
- 对于已发出 sqleseti 以设置客户机名称、应用程序名称、wkstn 或记帐的任何应用程序:
  - 添加 800 字节并加上记帐字符串本身的大小

相关概念:

- 第 3 页的『数据库系统监视器』
- 第 11 页的『系统监视开关』

相关任务:

- 第 12 页的『通过 CLP 设置监视开关』

相关参考:

- 『mon_heap_sz － 数据库系统监视器堆大小配置参数』（*《性能指南》*）

第 2 章 系统监视器开关

系统监视开关

收集系统监视器数据将引入处理数据库管理器的开销。例如，为了计算 SQL 语句的执行时间，数据库管理器必须对操作系统进行调用，以获取每个语句执行之前和执行之后的时间戳记。这些系统调用类型的成本通常很高。系统监视器导致的另一种形式的开销是增加内存消耗。对于系统监视器跟踪的每个监视元素，数据库管理器将使用内存来存储收集的数据。

为了将维护监视信息所涉及的开销降至最低，监视开关将控制数据库管理器可能进行的高成本信息收集。每个开关只有两个设置：ON 或 OFF。如果监视开关为 OFF，则受开关控制的监视元素不会收集任何信息。有很大一部分基本监视数据不受开关控制，并且不管开关设置如何，始终会收集这些信息。

每个监视应用程序都有自己的监视开关（和系统监视器数据）逻辑视图。在启动时，每个应用程序将从数据库管理器配置文件中的 dft_monswitches 参数继承其监视开关设置（在实例级别）。监视应用程序可使用 UPDATE MONITOR SWITCHES USING MONSWITCH OFF/ON 命令来改变其监视开关设置。MONSWITCH 参数包含下面的“快照监视开关”表的“监视开关”列中的值。在应用程序级别对开关设置的更改仅影响从中更改开关的应用程序。

可在不停止数据库管理系统的情况下更改实例级别的监视开关。为此，请使用 UPDATE DBM CFG USING DBMSWITCH OFF/ON 命令。DBMSWITCH 参数包含下面的“快照监视开关”表的“DBM 参数”列中的值。开关的动态更新要求执行更新的应用程序显式连接至实例，以使更改动态生效。动态更新不会影响其他现有快照应用程序。新的监视应用程序将继承已更新实例级别监视开关设置。要让现有监视应用程序继承新的缺省监视开关值，它必须终止并重新建立连接。如果在数据库管理器配置文件中更新开关，则会更新分区数据库中的所有分区的开关。

数据库管理器记录所有快照监视应用程序及其开关设置。如果开关在应用程序的配置中设置为 ON，则数据库管理器总是会收集该监视器数据。如果之后同一开关在应用程序配置中设置为 OFF，则只要至少有一个此开关设置为 ON 的应用程序，数据库管理器仍将收集数据。

时间和时间戳记元素的收集由 TIMESTAMP 开关控制。如果将此开关设置为 OFF（在缺省情况下设置为 ON），则指示在确定与时间或时间戳记相关的监视元素时，数据库管理器将跳过所有时间戳记操作系统调用。在 CPU 利用率达到 100% 时，应将此开关设置为 OFF。当此情况发生时，发出时间戳记导致的性能下降的幅度会急剧增长。对于可由 TIMESTAMP 开关和另一开关控制的监视元素，如果任一开关设置为 OFF，则不会收集数据。因此，如果 TIMESTAMP 开关设置为 OFF，则受另一监视开关控制的整体数据成本会大大降低。

事件监视器不会以与快照监视应用程序相同的方式受监视开关的影响。定义事件监视器时，它会将指定事件类型所需的实例级别监视开关自动设置为 ON。例如，死锁事件

监视器会将 LOCK 监视开关自动设置为 ON。在激活事件监视器时，必需的监视开关将设置为 ON。释放事件监视器时，监视开关将设置为 OFF。

事件监视器不会自动设置 TIMESTAMP 监视开关。这是控制所有监视元素的收集的唯一监视开关，这些监视元素属于事件监视器逻辑数据分组。如果 TIMESTAMP 开关设置为 OFF，则不会收集事件监视器收集的大多数时间戳记和时间监视元素。这些元素仍将写至指定的表、文件或管道，但必须带有值零。

表 2. 快照监视开关

监视开关	DBM 参数	提供的信息
BUFFERPOOL	DFT_MON_BUFPOOL	读写次数，所花时间
LOCK	DFT_MON_LOCK	锁定等待次数，死锁数
SORT	DFT_MON_SORT	使用的堆数，排序性能
STATEMENT	DFT_MON_STMT	启动／停止时间，语句标识
TABLE	DFT_MON_TABLE	活动量度（读／写行数）
UOW	DFT_MON_UOW	开始／结束时间，完成状态
TIMESTAMP	DFT_MON_TIMESTAMP	时间戳记

相关概念:

- 第 55 页的『事件监视器』
- 第 17 页的『监视开关自描述数据流』
- 第 19 页的『快照监视器』

相关任务:

- 第 14 页的『通过客户机应用程序设置监视开关』
- 第 12 页的『通过 CLP 设置监视开关』

通过 CLP 设置监视开关

在捕获快照或使用事件监视器之前，先确定需要数据库管理器收集的数据。如果想要收集下列任何特殊类型的数据，则需要设置监视开关。

- 缓冲池活动信息
- 锁定等待及与时间相关的锁定的信息
- 排序信息
- SQL 语句信息
- 表活动信息
- 时间和时间戳记信息
- 工作单元信息

在缺省情况下，与上述信息类型相对应的开关全部设置为“OFF”，但对应于时间和时间戳记信息的开关在缺省情况下设置为“ON”。

注: 事件监视器仅受时间和时间戳记信息开关影响。所有其他开关设置对事件监视器收集的数据没有影响。

先决条件:

执行任何监视开关更新的应用程序必须具有实例连接。

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限中的一种才能使用下列命令:

- UPDATE MONITOR SWITCHES
- GET MONITOR SWITCHES
- GET DATABASE MANAGER MONITOR SWITCHES

必须具有 SYSADM 权限才能使用 UPDATE DBM CFG 命令。

过程（UPDATE MONITOR SWITCHES）:

- 要激活任何局部监视开关，请使用 UPDATE MONITOR SWITCHES 命令。以下示例将所有局部监视开关更新为 ON:

  ```
  db2 update monitor switches using BUFFERPOOL on LOCK on
      SORT on STATEMENT on TIMESTAMP on TABLE on UOW on
  ```

  开关将保持活动状态，直到应用程序（CLP）拆离，或者直到再次使用 UPDATE MONITOR SWITCHES 命令释放它们。
- 要释放任何局部监视开关，请使用 UPDATE MONITOR SWITCHES 命令。以下示例将所有局部监视开关更新为 OFF:

  ```
  db2 update monitor switches using BUFFERPOOL off, LOCK off,
      SORT off, STATEMENT off, TIMESTAMP off, TABLE off, UOW off
  ```

  以下是在发出上述 UPDATE MONITOR SWITCH 命令后希望看到的输出的示例:

  ```
            监视器记录开关

  数据库分区号 1 的开关列表
  缓冲池活动信息（BUFFERPOOL） = OFF
  锁定信息（LOCK）             = OFF
  排序信息（SORT）             = OFF
  SQL 语句信息（STATEMENT）    = OFF
  表活动信息（TABLE）          = OFF
  工作单元信息（UOW）          = OFF
  获取时间戳记信息（TIMESTAMP）= OFF
  ```
- 还可以在数据库管理器级别操作监视开关。这涉及使用 UPDATE DBM CFG 命令在数据库管理器配置文件中更改 dft_monswitches 参数。

  ```
  db2 update dbm cfg using DFT_MON_LOCK on
  ```

  在上述示例中，除了基本信息之外，仅收集锁定开关控制的信息。

  每次启动监视应用程序时，它将从数据库管理器继承监视开关设置。对数据库管理器的监视开关所作的任何更改不会影响任何正在运行的监视应用程序。监视应用程序必须重新连接至实例以获取对监视开关设置的所有更改。
- 对于分区数据库系统，可专门为特定分区设置监视开关，或者为所有分区设置全局监视开关。要为特定分区（如分区号 3）设置监视开关（如 BUFFERPOOL），请发出以下命令:

  ```
  db2 update monitor switches using BUFFERPOOL on
      at dbpartitionnum 3
  ```

  要为所有分区设置监视开关（如 SORT），请发出以下命令:

  ```
  db2 update monitor switches using SORT on global
  ```

过程（GET MONITOR SWITCHES）:

- 要检查局部监视开关的状态，请使用 GET MONITOR SWITCHES 命令。

```
db2 get monitor switches
```

- 对于分区数据库系统，可专门查看特定分区的监视开关设置，或者查看所有分区的全局监视开关设置。要查看特定分区（如分区号 2）的监视开关设置，请发出以下命令:

```
db2 get monitor switches at dbpartitionnum 2
```

要查看所有分区的监视开关设置，请发出以下命令:

```
db2 get monitor switches global
```

过程（GET DATABASE MANAGER MONITOR SWITCHES）:

- 要在数据库管理器级别（或实例级别）检查监视开关的状态，请使用 GET DATABASE MANAGER MONITOR SWITCHES 命令。此命令将显示要监视的实例的整体开关设置。

```
db2 get database manager monitor switches
```

以下是发出上述命令后希望看到的输出的示例:

```
          收集的 DBM 系统监视器信息

数据库分区号 1 的开关列表
缓冲池活动信息（BUFFERPOOL）   = OFF
锁定信息（LOCK）               = ON    10-25-2001 16:04:39
排序信息（SORT）               = OFF
SQL 语句信息（STATEMENT）      = OFF
表活动信息（TABLE）            = OFF
工作单元信息（UOW）            = OFF
获取时间戳记信息（TIMESTAMP）  = OFF
```

既然已经设置了期望的监视开关并且确认了开关设置，就可以捕获并收集监视器数据了。

相关概念:

- 第 55 页的『事件监视器』
- 第 19 页的『快照监视器』
- 第 11 页的『系统监视开关』

相关任务:

- 第 14 页的『通过客户机应用程序设置监视开关』

通过客户机应用程序设置监视开关

在捕获快照或使用事件监视器之前，必须确定需要数据库管理器收集的数据。如果想要收集下列任何特殊类型的数据，则需要设置适当的监视开关。

- 缓冲池活动信息
- 锁定、锁定等待及与时间有关的锁定信息
- 排序信息
- SQL 语句信息
- 表活动信息
- 时间和时间戳记信息

• 工作单元信息

在缺省情况下，与上述信息类型相对应的开关全部设置为“OFF”，但对应于时间和时间戳记信息的开关在缺省情况下设置为“ON”。

注: 事件监视器仅受时间和时间戳记信息开关影响。所有其他开关设置对事件监视器收集的数据没有影响。

先决条件:

执行任何监视开关更新的应用程序必须具有实例连接。

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能使用 db2MonitorSwitches API。

过程:

1. 包括下列 DB2 库：sqlutil.h 和 db2ApiDf.h。可在 `sqllib` 下的 `include` 子目录中找到它们。

```
#include <sqlutil.h>
#include <db2ApiDf.h>
#include <string.h>
#include <sqlmon.h>
```

2. 将开关列表缓冲区单元大小设置为 1 KB。

```
#define SWITCHES_BUFFER_UNIT_SZ 1024
```

3. 初始化 sqlca、db2MonitorSwitches 和 sqlm_recording_group 结构。还应初始化指针以包含开关列表缓冲区并确定缓冲区大小。

```
struct sqlca sqlca;
memset (&sqlca, '\0', sizeof(struct sqlca));
db2MonitorSwitchesData switchesData;
memset (&switchesData, '\0', sizeof(switchesData));
struct sqlm_recording_group switchesList[SQLM_NUM_GROUPS];
memset(switchesList, '\0', sizeof(switchesList));
sqluint32 outputFormat;
static sqluint32 switchesBufferSize = SWITCHES_BUFFER_UNIT_SZ;
char *switchesBuffer;
```

4. 初始化缓冲区以容纳开关列表输出。

```
switchesBuffer = (char *)malloc(switchesBufferSize);
memset(switchesBuffer, '\0', switchesBufferSize));
```

5. 要改变局部监视开关的状态，请改变 sqlm_recording_group 结构中的元素（按先前步骤中的指示命名为 switchesList）。对于要设置为“ON”的监视开关，参数 input_state 应设置为 SQLM_ON。对于要设置为“OFF”的监视开关，参数 input_state 必须设置为 SQLM_OFF。

```
switchesList[SQLM_UOW_SW].input_state = SQLM_ON;
switchesList[SQLM_STATEMENT_SW].input_state = SQLM_ON;
switchesList[SQLM_TABLE_SW].input_state = SQLM_ON;
switchesList[SQLM_BUFFER_POOL_SW].input_state = SQLM_OFF;
switchesList[SQLM_LOCK_SW].input_state = SQLM_OFF;
switchesList[SQLM_SORT_SW].input_state = SQLM_OFF;
switchesList[SQLM_TIMESTAMP_SW].input_state = SQLM_OFF;
switchesData.piGroupStates = switchesList;
switchesData.poBuffer = switchesBuffer;
switchesData.iVersion = SQLM_DBMON_VERSION8;
switchesData.iBufferSize = switchesBufferSize;
```

```
switchesData.iReturnData = 0;
switchesData.iNodeNumber = SQLM_CURRENT_NODE;
switchesData.poOutputFormat = &outputFormat;
```

注意，如果 iVersion 小于 SQLM_DBMON_VERSION8，则 SQLM_TIMESTAMP_SW 不可用。

6. 要提交对开关设置的更改，请调用 db2MonitorSwitches() 函数。将 db2MonitorSwitchesData 结构（在此示例中命名为 switchesData）作为参数传递至 db2MonitorSwitches API。switchesData 以参数的形式包含 sqlm_recording_group 结构。

   ```
   db2MonitorSwitches(db2Version810, &switchesData, &sqlca);
   ```

7. 从开关列表缓冲区处理开关列表数据流。
8. 清除开关列表缓冲区。

   ```
   free(switchesBuffer);
   free(pRequestedDataGroups);
   ```

既然已经设置了期望的监视开关并且确认了开关设置，就可以捕获并收集监视器数据了。

相关概念:

- 第 55 页的『事件监视器』
- 第 17 页的『监视开关自描述数据流』
- 第 19 页的『快照监视器』
- 第 11 页的『系统监视开关』

相关任务:

- 第 12 页的『通过 CLP 设置监视开关』

相关参考:

- 『db2MonitorSwitches API - Get or update the monitor switch settings』（*Administrative API Reference*）

相关样本:

- 『dbsnap.cbl -- Get a database monitor snapshot (IBM COBOL)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C)』
- 『clisnap.c -- Capture a snapshot at the client level (C)』
- 『dbsnap.c -- Capture a snapshot at the database level (C)』
- 『insnap.c -- Capture a snapshot at the instance level (C)』
- 『utilsnap.c -- Utilities for the snapshot monitor samples (C)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C++)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C++)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C++)』
- 『clisnap.C -- Capture a snapshot at the client level (C++)』
- 『dbsnap.C -- Capture a snapshot at the database level (C++)』
- 『insnap.C -- Capture a snapshot at the instance level (C++)』

- 『utilsnap.C -- Utilities for the snapshot monitor samples (C++)』

监视开关自描述数据流

在使用 db2MonitorSwitches API 更新或查看当前监视开关设置之后，API 将以自描述数据流的形式返回开关设置。图 1 显示可能对分区数据库环境返回的开关列表信息的结构。

注:

1. 描述性名称用于示例和表中的标识。在实际数据流中，将在这些名称前加上 **SQLM_ELM_** 前缀。例如，db_event 在事件监视器输出中显示为 SQLM_ELM_DB_EVENT。在实际数据流中，将在类型前加上 **SQLM_TYPE_** 前缀。例如，HEADER 在数据流中将显示为 SQLM_TYPE_HEADER。
2. 对于全局开关请求，返回的信息的分区顺序在每个开关请求中可能有所不同。在此情况下，分区标识包括在数据流中。

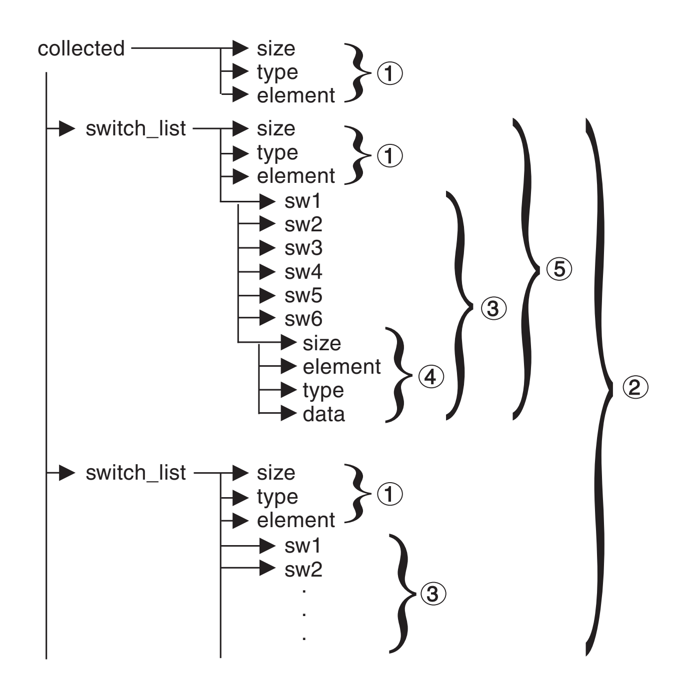


图 1. 开关列表监视器数据流

1. 每个逻辑数据组以指示其大小和名称的头开始。此大小不包括头本身占用的数据量。
2. 已收集头中的大小返回所有分区的所有监视开关列表的总大小。
3. 开关列表头中的大小元素指示该分区的开关数据的大小。
4. 开关信息是自描述格式。
5. 对于非分区数据库，将返回独立分区的开关设置。即只返回一个开关列表。

相关概念:

- 第 7 页的『系统监视器输出：自描述数据流』
- 第 11 页的『系统监视开关』

相关任务:

- 第 14 页的『通过客户机应用程序设置监视开关』

相关参考:

- 『db2MonitorSwitches API - Get or update the monitor switch settings』（*Administrative API Reference*）

第 3 章 使用快照监视器

快照监视器

可使用快照监视器以在特定时间捕获有关数据库和所有已连接应用程序的信息。快照对于确定数据库系统的状态非常有用。如果每隔一定时间间隔获取快照，则快照在观察趋势和预测潜在问题方面时也很有用。要获取给定时间段的所有数据库的监视器信息，请使用事件监视器。

系统监视器仅累积数据库处于活动状态时的信息。如果所有应用程序与数据库断开连接并且数据库释放，则不再提供该数据库的系统监视器数据。可通过使用 ACTIVATE DATABASE 命令启动数据库或保持与数据库的永久连接，以便让数据库保持活动状态直到获取最终快照。

快照监视需要实例连接。如果没有实例连接，则创建缺省实例连接。通常，对于应用程序调用第一个数据库系统监视器 API 时，将隐式建立与 DB2INSTANCE 环境变量指定的实例的连接。还可使用 ATTACH TO 命令显式建立连接。一旦连接了应用程序，它调用的所有系统监视器请求都将引导至该实例。这允许客户机只需要连接至远程服务器上的实例就可以监视该远程服务器。

在分区数据库环境中，可在实例的任何分区上获取快照，或者使用单个实例连接获取全局快照。全局快照聚集在每个分区上收集到的数据并返回一组值。

可从 CLP、SQL 表函数捕获快照，也可以通过使用 C 或 C++ 应用程序中的快照监视器 API 来捕获快照。有若干不同快照请求类型可用，每种类型返回特定类型的监视数据。例如，可捕获仅返回缓冲池信息的快照，或者捕获返回数据库管理器信息的快照。在捕获快照前，请考虑是否需要由监视开关控制的监视元素提供的信息。如果特定监视开关处于关闭状态，就不会收集它控制的监视元素。

相关概念:

- 第 3 页的『数据库系统监视器』
- 第 3 页的『数据库系统监视器数据结构』
- 第 48 页的『分区数据库系统上的全局快照』
- 第 47 页的『子节快照』
- 第 11 页的『系统监视开关』

相关任务:

- 第 41 页的『从客户机应用程序捕获数据库快照』
- 第 39 页的『从 CLP 捕获数据库快照』
- 第 52 页的『对数据库系统快照的 SQL 访问』

相关参考:

- 第 45 页的『快照监视器样本输出』

访问系统监视器数据：SYSMON 权限

作为 SYSMON 数据库管理器级别组的成员的用户有权获取对数据库系统监视器数据的访问权。系统监视器数据是通过使用快照监视器 API、CLP 命令或 SQL 表函数来访问的。

SYSMON 权限组替换 DB2_SNAPSHOT_NOAUTH 注册表变量以使没有系统管理或系统控制权限的用户能够访问数据库系统监视器。

除了 SYSMON 权限之外，访问使用快照监视器的系统监视器数据的唯一方法是使用系统管理或系统控制权限。

属于 SYSMON 组或具有系统管理或系统控制权限的任何用户可以执行下列快照监视器函数:

- CLP 命令:
  - GET DATABASE MANAGER MONITOR SWITCHES
  - GET MONITOR SWITCHES
  - GET SNAPSHOT
  - LIST ACTIVE DATABASES
  - LIST APPLICATIONS
  - LIST DCS APPLICATIONS
  - RESET MONITOR
  - UPDATE MONITOR SWITCHES
- API:
  - db2GetSnapshot － 获取快照
  - db2GetSnapshotSize － 估计 *db2GetSnapshot()* 输出缓冲区所需的大小
  - db2MonitorSwitches － 获取／更新监视开关
  - db2ResetMonitor － 复位监视器
- 无需预先运行 SYSPROC.SNAP_WRITE_FILE 的快照 SQL 表函数

相关概念:

- 『系统监视器权限（SYSMON）』（*《管理指南: 实施》*）
- 第 19 页的『快照监视器』

使用快照管理视图和表函数来捕获数据库系统快照

授权用户可使用快照管理视图或快照表函数来捕获 DB2 实例的监视器信息快照。快照管理视图提供了一种简单的方法，可用来访问已连接数据库的所有数据库分区的数据。快照表函数允许您请求特定数据库分区的数据，全局聚集数据或所有数据库分区中的数据。某些快照表函数允许您请求所有活动数据库中的数据。

虽然新的监视器数据可用时在将来发行版中可能需要新的快照表函数，但在新列添加至视图时快照管理视图集合将保持不变，使得长期进行应用程序维护时管理视图成为一个很好的选择。

每个快照视图返回一个表，每个数据库分区每个被监视对象对应一行，每列表示一个监视元素。每个表函数返回一个表，指定分区的每个被监视对象对应一行。返回的表的列名与监视元素名称相关。

例如，SAMPLE 数据库的一般应用程序信息的快照将使用如下 SNAPAPPL 管理视图进行捕获:

```
SELECT * FROM SYSIBMADM.SNAPAPPL
```

您也可以从返回的表中选择个别监视元素。例如，以下语句仅返回 agent_id 和 appl_id 监视元素:

```
SELECT agent_id, appl_id FROM SYSIBMADM.SNAPAPPL
```

先决条件:

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能捕获数据库快照。

要获取远程实例的快照，必须先连接至属于该实例的本地数据库。

限制:

快照管理视图和表函数不能与下列任一项一起使用:

- 监视开关命令 / API
- 监视器复位命令 / API

此限制包括:

- GET MONITOR SWITCHES
- UPDATE MONITOR SWITCHES
- RESET MONITOR

此限制是由于此类命令使用 INSTANCE ATTACH，而快照表函数使用 DATABASE CONNECT 造成的。

过程:

要使用快照管理视图捕获快照，您必须:

1. 连接到数据库。这可以是实例中任何需要监视的数据库。为了能够使用快照管理视图发出 SQL 查询，必须连接到数据库。
2. 确定需要捕获的快照的类型。如果想要捕获当前连接的数据库之外的数据库的快照，或者如果想要从单个数据库分区或全局聚集数据中检索数据，则需要改为使用快照表函数。
3. 使用适当的快照管理视图发出查询。例如，以下查询将对当前连接的数据库捕获锁定信息的快照:

   ```
   SELECT * FROM SYSIBMADM.SNAPLOCK
   ```

要使用快照表函数捕获快照，您必须:

1. 连接到数据库。这可以是实例中任何需要监视的数据库。为了能够使用快照表函数发出 SQL 查询，必须连接到数据库。
2. 确定需要捕获的快照的类型。
3. 使用适当的快照表函数发出查询。例如，以下查询将对当前连接的数据库分区捕获有关 SAMPLE 数据库的锁定信息的快照:

```
SELECT * FROM TABLE(SNAP_GET_LOCK('SAMPLE',-1)) AS SNAPLOCK
```

SQL 表函数有两个输入参数:

数据库名称
VARCHAR(255)。如果输入 NULL，就会使用当前连接的数据库的名称。

分区号 SMALLINT。对于数据库分区号参数，输入对应于需要监视的数据库分区号的整数（0 到 999 之间的值）。要捕获当前连接的数据库分区的快照，请输入值 -1。要捕获全局聚集快照，请输入值 -2。要从所有数据库分区捕获快照，不要对此参数指定值。

注:

a. 对于以下快照表函数列表，如果对当前连接的数据库输入 NULL，则将获取实例中所有数据库的快照信息:
   - SNAP_GET_DB_V91
   - SNAP_GET_DB_MEMORY_POOL
   - SNAP_GET_DETAILLOG_V91
   - SNAP_GET_HADR
   - SNAP_GET_STORAGE_PATHS
   - SNAP_GET_APPL
   - SNAP_GET_APPL_INFO
   - SNAP_GET_AGENT
   - SNAP_GET_AGENT_MEMORY_POOL
   - SNAP_GET_STMT
   - SNAP_GET_SUBSECTION
   - SNAP_GET_BP
   - SNAP_GET_BP_PART

b. 数据库名称参数不适用于数据库管理器级别快照表函数；那些表函数只有数据库分区号参数。数据库分区号参数是可选的。

相关概念:

- 第 19 页的『快照监视器』
- 『Administrative SQL routines and views』（*Administrative SQL Routines and Views*）

相关参考:

- 第 93 页的『快照监视器接口至逻辑数据组的映射』
- 第 34 页的『快照监视器 SQL 管理视图』
- 『Administrative views versus table functions』（*Administrative SQL Routines and Views*）

使用 SNAP_WRITE_FILE 存储过程来将数据库系统快照信息捕获到文件中

通过使用 SNAP_WRITE_FILE 存储过程，可以捕获监视器数据快照并将此信息存储到数据库服务器上的文件中，并且可以允许未拥有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限的用户访问该数据。于是，任何用户都可以发出带有快照表函数的查询以访问这些文件中的快照信息。如果开放对快照监视器数据的访问，所有对快照表函数

具有执行特权的用户就能获得一些敏感信息，例如已连接用户列表以及他们对数据库提交的 SQL 语句等。缺省情况下，已将执行快照表函数的特权授予 PUBLIC。

注: 使用快照监视器表函数并不会泄漏表中的实际数据或用户密码。

调用 SNAP_WRITE_FILE 存储过程时，除了标识所要监视的数据库和分区以外，还需要指定*快照请求类型*。每种快照请求类型都确定了所收集的监视器数据的范围。请根据用户需要运行的快照表函数来选择快照请求类型。下表列示了快照表函数及其对应的请求类型。

表 3. 快照请求类型

快照表函数	快照请求类型
SNAP_GET_AGENT	APPL_ALL
SNAP_GET_AGENT_MEMORY_POOL	APPL_ALL
SNAP_GET_APPL	APPL_ALL
SNAP_GET_APPL_INFO	APPL_ALL
SNAP_GET_STMT	APPL_ALL
SNAP_GET_SUBSECTION	APPL_ALL
SNAP_GET_BP_PART	BUFFERPOOLS_ALL
SNAP_GET_BP	BUFFERPOOLS_ALL
SNAP_GET_DB_V91	DBASE_ALL
SNAP_GET_DETAILLOG_V91	DBASE_ALL
SNAP_GET_DB_MEMORY_POOL	DBASE_ALL
SNAP_GET_HADR	DBASE_ALL
SNAP_GET_STORAGE_PATHS	DBASE_ALL
SNAP_GET_DBM	DB2
SNAP_GET_DBM_MEMORY_POOL	DB2
SNAP_GET_FCM	DB2
SNAP_GET_FCM_PART	DB2
SNAP_GET_SWITCHES	DB2
SNAP_GET_DYN_SQL_V91	DYNAMIC_SQL
SNAP_GET_LOCK	DBASE_LOCKS
SNAP_GET_LOCKWAIT	APPL_ALL
SNAP_GET_TAB_V91	DBASE_TABLES
SNAP_GET_TAB_REORG	DBASE_TABLES
SNAP_GET_TBSP_V91	DBASE_TABLESPACES
SNAP_GET_TBSP_PART_V91	DBASE_TABLESPACES
SNAP_GET_CONTAINER_V91	DBASE_TABLESPACES
SNAP_GET_TBSP_QUIESCER	DBASE_TABLESPACES
SNAP_GET_TBSP_RANGE	DBASE_TABLESPACES
SNAP_GET_UTIL	DB2
SNAP_GET_UTIL_PROGRESS	DB2

先决条件:

您必须拥有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能使用 SNAP_WRITE_FILE 存储过程来捕获数据库快照。

过程:

要使用 SNAP_WRITE_FILE 存储过程来将快照捕获到文件中:

1. 连接到数据库。这可以是实例中任何需要监视的数据库。为了能够调用存储过程，必须连接到数据库。
2. 确定快照请求类型以及需要监视的数据库和分区。
3. 调用 SNAP_WRITE_FILE 存储过程，对快照请求类型参数、数据库参数和分区参数指定适当的设置。例如，以下调用将捕获 SAMPLE 数据库的当前连接分区的应用程序信息快照:

   ```
   CALL SNAP_WRITE_FILE('APPL_ALL','SAMPLE',-1)
   ```

   SNAP_WRITE_FILE 存储过程有 3 个输入参数:

   - 快照请求类型（请参阅第 23 页的表 3，该表提供了快照表函数及其相应请求类型的交叉引用）
   - VARCHAR (128) 数据库名称。如果输入 NULL，就会使用当前连接的数据库的名称。

     注: 此参数不适用于数据库管理器级别快照表函数；那些表函数只有请求类型参数和分区号参数。
   - SMALLINT 分区号（0 到 999 之间的值）。对于分区号参数，输入与所要监视的分区号相对应的整数。要捕获当前连接的分区的快照，请输入值 -1 或 NULL。要捕获全局快照，请输入值 -2。

在将快照数据保存到文件中之后，通过将 (NULL, NULL) 指定为数据库级别表函数的输入值并对数据库管理器级别表函数指定 (NULL)，所有用户都可以发出包含相应快照表函数的查询。他们接收到的监视器数据是从 SNAP_WRITE_FILE 存储过程生成的文件中得来的。

虽然这种方法限制了用户对敏感监视器数据的访问，但这种方法有一些局限性:

- SNAP_WRITE_FILE 文件中的快照监视器数据仅仅是上次调用 SNAP_WRITE_FILE 存储过程时的最新数据。通过定期调用 SNAP_WRITE_FILE 存储过程，可以确保获得最新的快照监视器数据。例如，在 UNIX® 系统上，可以设置 cron 作业来完成此操作。
- 发出包含快照表函数的查询的用户不能标识所要监视的数据库或分区。发出 SNAP_WRITE_FILE 调用的用户所标识的数据库名称和分区号确定了快照表函数可访问的文件内容。
- 如果用户发出包含快照表函数的 SQL 查询，但尚未针对该快照表函数运行相应的 SNAP_WRITE_FILE 请求类型，就会尝试对当前连接的数据库和分区创建直接快照。仅当用户具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限时，此操作才会成功。

相关概念:

- 第 19 页的『快照监视器』

相关任务:

- 第 25 页的『使用 SQL 查询中的快照表函数来访问数据库系统快照（使用文件访问）』

相关参考:

- 第 93 页的『快照监视器接口至逻辑数据组的映射』
- 第 34 页的『快照监视器 SQL 管理视图』
- 『SNAP_WRITE_FILE procedure』（*Administrative SQL Routines and Views*）

使用 SQL 查询中的快照表函数来访问数据库系统快照（使用文件访问）

对于授权用户调用 SNAP_WRITE_FILE 存储过程的每个请求类型，任何用户都可使用相应快照表函数发出查询。他们接收到的监视器数据是从 SNAP_WRITE_FILE 存储过程生成的文件中检索到的。

用户通过在 SQL 查询中使用快照表函数以从 SNAP_WRITE_FILE 文件访问快照数据。例如，SAMPLE 数据库的一般应用程序信息的快照将为如下所示:

```
SELECT * FROM TABLE( SNAP_GET_APPL(CAST(NULL AS VARCHAR(1)),
                                   CAST (NULL AS INTEGER)))
                     as SNAP_GET_APPL
```

每个快照表函数返回带有一行或多行的一个表，每列表示一个监视元素。相应的，监视元素列名与监视元素名称相关。

您也可以从返回的表中选择个别监视元素。例如，以下语句仅返回 agent_id 监视元素:

```
SELECT agent_id FROM TABLE(
                           SNAP_GET_APPL(CAST(NULL AS VARCHAR(1)),
                                         CAST (NULL AS INTEGER)))
                           as SNAP_GET_APPL
```

先决条件:

对于打算用于访问 SNAP_WRITE_FILE 文件的每个快照表函数，授权用户必须已经使用相应快照请求类型发出了 SNAP_WRITE_FILE 存储过程调用。

限制:

使用快照表函数从 SNAP_WRITE_FILE 文件访问快照数据的用户不能标识要监视的数据库或分区。如果数据库名称和分区号是通过用户发出 SNAP_WRITE_FILE 调用标识的，则该数据库名称和分区号将确定 SNAP_WRITE_FILE 文件的内容。

SNAP_WRITE_FILE 文件中可用的快照监视器数据仅仅是上次 SNAP_WRITE_FILE 存储过程捕获快照时的最新数据。

如果发出包含快照表函数的 SQL 查询，但尚未针对该快照表函数运行相应的 SNAP_WRITE_FILE 请求类型，就会尝试对当前连接的数据库和分区创建直接快照。仅当用户具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限时，此操作才会成功。

过程:

要使用快照表函数从 SNAP_WRITE_FILE 文件访问快照数据，必须:

1. 连接到数据库。这可以是实例中任何需要监视的数据库。要使用快照表函数发出 SQL 查询，必须连接到数据库。
2. 确定需要捕获的快照的类型。
3. 使用适当的快照表函数发出查询。例如，以下查询将捕获表空间信息的快照。

```
SELECT * FROM TABLE(SNAP_GET_TBSP_V91 (CAST(NULL AS VARCHAR(1)),
                    CAST (NULL AS INTEGER))) AS SNAP_GET_TBSP_V91
```

注： 必须对数据库名称和分区号参数输入空值。快照的数据库名称和分区将在 SNAP_WRITE_FILE 存储过程的调用中确定。而且，数据库名称参数不适用于数据库管理器级别快照表函数；那些表函数只有分区号参数。

相关概念:

- 第 19 页的『快照监视器』

相关任务:

- 第 52 页的『对数据库系统快照的 SQL 访问』
- 第 22 页的『使用 SNAP_WRITE_FILE 存储过程来将数据库系统快照信息捕获到文件中』
- 第 20 页的『使用快照管理视图和表函数来捕获数据库系统快照』

相关参考:

- 第 93 页的『快照监视器接口至逻辑数据组的映射』
- 第 34 页的『快照监视器 SQL 管理视图』

使用快照监视器数据来监视分区表的重组

没有单独的用于指示分区表整体表重组状态的数据组。分区表使用了数据组织方案，即，表数据根据该表中一个或多个表分区键列中的值分布到多个存储对象（称为数据分区或范围）中。但是，可以根据所重组的各个数据分区数据组中的元素值来推断全局表重组状态。下列信息描述了一些最有用的全局表重组状态监视方法。

确定重组的数据分区数:

通过计算表名和模式名相同的表数据监视器数据块数，可以确定表中重组的数据分区总数。此值指示启动了重组的数据分区数。示例 1 和 2 指示正在对 3 个数据分区进行重组。

标识所重组的数据分区:

可以根据阶段开始时间（reorg_phase_start）来推断当前正在重组的数据分区。在 SORT/BUILD/REPLACE 阶段，与正在重组的数据分区相对应的监视器数据显示了最新阶段开始时间。在 INDEX_RECREATE 阶段，所有数据分区的阶段开始时间都是相同的。在示例 1 和 2 中，指示了 INDEX_RECREATE 阶段，因此所有数据分区的开始时间都是相同的。

标识索引重建需求:

通过获取与任何一个正在重组的数据分区相对应的最大重组阶段数（reorg_max_phase）元素值，可以确定是否需要重建索引。如果 reorg_max_phase 值为 3 或 4，则表示需要重建索引。示例 1 和 2 报告的 reorg_max_phase 值为 3，即表示需要重建索引。

以下样本输出来自一台 3 节点服务器，该服务器包含一个带有 3 个数据分区的表:

```
CREATE TABLE  sales (c1 INT, c2 INT, c3 INT)
    PARTITION BY RANGE (c1)
        (PART P1 STARTING FROM (1)  ENDING AT (10) IN parttbs,
         PART P2 STARTING FROM (11) ENDING AT (20) IN parttbs,
         PART P3 STARTING FROM (21) ENDING AT (30) IN parttbs)
    DISTRIBUTE BY (c2)
```

执行的语句:

```
REORG TABLE sales ALLOW NO ACCESS ON ALL DBPARTITIONNUMS
```

示例 1:

```
GET SNAPSHOT FOR TABLES ON DPARTDB GLOBAL
```

已将输出修改为仅包括相关表的表信息。

```
            表快照

第一个数据库连接时间戳记                 = 06/28/2005 13:46:43.061690
上次复位时间戳记                         = 06/28/2005 13:46:47.440046
快照时间戳记                             = 06/28/2005 13:46:50.964033
数据库名称                               = DPARTDB
数据库路径                               = /work/sales/NODE0000/SQL00001/
输入数据库别名                           = DPARTDB
已访问的表的数目                         = 5

表列表
 表模式                  = NEWTON
 表名                    = SALES
 表类型                  = 用户
 数据分区标识            = 0
 数据对象页              = 3
 读取的行数              = 12
 写入的行数              = 1
 溢出数                  = 0
 重组的页数              = 0
 表重组信息:
   节点数                = 0
   重组类型              =
        回收
        表重组
        不允许访问
        通过表扫描重新集群
        仅重组数据
   重组索引数            = 0
   重组表空间数          = 3
 长临时空间标识          = 3
   开始时间              = 06/28/2005 13:46:49.816883
   重组阶段              = 3 - 索引重建
   最大阶段              = 3
   阶段开始时间          = 06/28/2005 13:46:50.362918
   状态                  = 已完成
   当前计数器            = 0
   最大计数器            = 0
   完成                  = 0
   结束时间              = 06/28/2005 13:46:50.821244
```

```
表重组信息:
  节点数              = 1
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:49.822701
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.420741
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.899543


表重组信息:
  节点数              = 2
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:49.814813
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.344277
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.803619


表模式                = NEWTON
表名                  = SALES
表类型                = 用户
数据分区标识          = 1
数据对象页            = 3
读取的行数            = 8
写入的行数            = 1
溢出数                = 0
重组的页数            = 0
表重组信息:
  节点数              = 0
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:50.014617
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
```

```
  阶段开始时间       = 06/28/2005 13:46:50.362918
  状态               = 已完成
  当前计数器         = 0
  最大计数器         = 0
  完成               = 0
  结束时间           = 06/28/2005 13:46:50.821244

表重组信息:
  节点数             = 1
  重组类型           =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数         = 0
  重组表空间数       = 3
长临时空间标识       = 3
  开始时间           = 06/28/2005 13:46:50.026278
  重组阶段           = 3 - 索引重建
  最大阶段           = 3
  阶段开始时间       = 06/28/2005 13:46:50.420741
  状态               = 已完成
  当前计数器         = 0
  最大计数器         = 0
  完成               = 0
  结束时间           = 06/28/2005 13:46:50.899543

表重组信息:
  节点数             = 2
  重组类型           =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数         = 0
  重组表空间数       = 3
长临时空间标识       = 3
  开始时间           = 06/28/2005 13:46:50.006392
  重组阶段           = 3 - 索引重建
  最大阶段           = 3
  阶段开始时间       = 06/28/2005 13:46:50.344277
  状态               = 已完成
  当前计数器         = 0
  最大计数器         = 0
  完成               = 0
  结束时间           = 06/28/2005 13:46:50.803619


表模式               = NEWTON
表名                 = SALES
表类型               = 用户
数据分区标识         = 2
数据对象页           = 3
读取的行数           = 4
写入的行数           = 1
溢出数               = 0
重组的页数           = 0
表重组信息:
  节点数             = 0
  重组类型           =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
```

```
  重组索引数          = 0
  重组表空间数        = 3
 长临时空间标识       = 3
  开始时间            = 06/28/2005 13:46:50.199971
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.362918
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.821244

 表重组信息:
  节点数              = 1
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
 长临时空间标识       = 3
  开始时间            = 06/28/2005 13:46:50.223742
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.420741
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.899543

 表重组信息:
  节点数              = 2
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
 长临时空间标识       = 3
  开始时间            = 06/28/2005 13:46:50.179922
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.344277
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.803619
```

示例 2:

```
GET SNAPSHOT FOR TABLES ON DPARTDB AT DBPARTITIONNUM 2
```

已将输出修改为仅包括相关表的表信息。

```
            表快照

第一个数据库连接时间戳记                = 06/28/2005 13:46:43.617833
上次复位时间戳记                        =
快照时间戳记                            = 06/28/2005 13:46:51.016787
```

```
数据库名称                          = DPARTDB
数据库路径                          = /work/sales/NODE0000/SQL00001/
输入数据库别名                      = DPARTDB
已访问的表的数目                    = 3

表列表
 表模式                 = NEWTON
 表名                   = SALES
 表类型                 = 用户
 数据分区标识           = 0
 数据对象页             = 1
 读取的行数             = 0
 写入的行数             = 0
 溢出数                 = 0
 重组的页数             = 0
 表重组信息:
   节点数               = 2
   重组类型             =
        回收
        表重组
        不允许访问
        通过表扫描重新集群
        仅重组数据
   重组索引数           = 0
   重组表空间数         = 3
 长临时空间标识         = 3
   开始时间             = 06/28/2005 13:46:49.814813
   重组阶段             = 3 - 索引重建
   最大阶段             = 3
   阶段开始时间         = 06/28/2005 13:46:50.344277
   状态                 = 已完成
   当前计数器           = 0
   最大计数器           = 0
   完成                 = 0
   结束时间             = 06/28/2005 13:46:50.803619


 表模式                 = NEWTON
 表名                   = SALES
 表类型                 = 用户
 数据分区标识           = 1
 数据对象页             = 1
 读取的行数             = 0
 写入的行数             = 0
 溢出数                 = 0
 重组的页数             = 0
 表重组信息:
   节点数               = 2
   重组类型             =
        回收
        表重组
        不允许访问
        通过表扫描重新集群
        仅重组数据
   重组索引数           = 0
   重组表空间数         = 3
 长临时空间标识         = 3
   开始时间             = 06/28/2005 13:46:50.006392
   重组阶段             = 3 - 索引重建
   最大阶段             = 3
   阶段开始时间         = 06/28/2005 13:46:50.344277
   状态                 = 已完成
   当前计数器           = 0
   最大计数器           = 0
   完成                 = 0
   结束时间             = 06/28/2005 13:46:50.803619
```

```
 表模式               = NEWTON
 表名                 = SALES
 表类型               = 用户
 数据分区标识         = 2
 数据对象页           = 1
 读取的行数           = 4
 写入的行数           = 1
 溢出数               = 0
 重组的页数           = 0
 表重组信息:
   节点数             = 2
   重组类型           =
        回收
        表重组
        不允许访问
        通过表扫描重新集群
        仅重组数据
   重组索引数         = 0
   重组表空间数       = 3
 长临时空间标识       = 3
   开始时间           = 06/28/2005 13:46:50.179922
   重组阶段           = 3 - 索引重建
   最大阶段           = 3
   阶段开始时间       = 06/28/2005 13:46:50.344277
   状态               = 已完成
   当前计数器         = 0
   最大计数器         = 0
   完成               = 0
   结束时间           = 06/28/2005 13:46:50.803619
```

示例 3:

SELECT * FROM SYSIBMADM.SNAPLOCK WHERE tabname = 'SALES';

已将输出修改为仅包括相关表的表信息的子集。

```
 ...   TBSP_NAME TABNAME LOCK_OBJECT_TYPE   LOCK_MODE  LOCK_STATUS ...
       --------- ------- ------------------ ---------- -----------
 ...   PARTTBS   SALES      ROW_LOCK        X          GRNT        ...
 ...   -         SALES      TABLE_LOCK      IX         GRNT        ...
 ...   PARTTBS   SALES      TABLE_PART_LOCK IX         GRNT        ...
 ...   PARTTBS   SALES      ROW_LOCK        X          GRNT        ...
 ...   -         SALES      TABLE_LOCK      IX         GRNT        ...
 ...   PARTTBS   SALES      TABLE_PART_LOCK IX         GRNT        ...
 ...   PARTTBS   SALES      ROW_LOCK        X          GRNT        ...
 ...   -         SALES      TABLE_LOCK      IX         GRNT        ...
 ...   PARTTBS   SALES      TABLE_PART_LOCK IX         GRNT        ...
```

选择了 9 个记录。

此查询（已继续）的输出。

```
 ... LOCK_ESCALATION LOCK_ATTRIBUTES DATA_PARTITION_ID DBPARTITIONNUM
     --------------- --------------- ----------------- --------------
 ...               0 INSERT          2                 2
 ...               0 NONE            -                 2
 ...               0 NONE            2                 2
 ...               0 INSERT          0                 0
 ...               0 NONE            -                 0
 ...               0 NONE            0                 0
 ...               0 INSERT          1                 1
 ...               0 NONE            -                 1
 ...               0 NONE            1                 1
```

示例 4:

SELECT * FROM SYSIBMADM.SNAPTAB WHERE tabname = 'SALES';

已将输出修改为仅包括相关表的表信息的子集。

```
... TABSCHEMA TABNAME TAB_FILE_ID TAB_TYPE   DATA_OBJECT_PAGES ROWS_WRITTEN ...
... --------- ------- ----------- ---------- ----------------- ------------ ...
... NEWTON    SALES             2 USER_TABLE                 1            1 ...
... NEWTON    SALES             4 USER_TABLE                 1            1 ...
... NEWTON    SALES             3 USER_TABLE                 1            1 ...

     选择了 3 个记录。
```

此查询（已继续）的输出。

```
... OVERFLOW_ACCESSES PAGE_REORGS DBPARTITIONNUM TBSP_ID DATA_PARTITION_ID
... ----------------- ----------- -------------- ------- -----------------
...                 0           0              0       3                 0
...                 0           0              2       3                 2
...                 0           0              1       3                 1
```

示例 5:

SELECT * FROM SYSIBMADM.SNAPTAB_REORG WHERE tabname = 'SALES';;

已将输出修改为仅包括相关表的表信息的子集。

```
  REORG_PHASE   REORG_MAX_PHASE REORG_TYPE                                    ...
     ------------- --------------- ------------------------------------------
 INDEX_RECREATE               3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE               3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE               3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE               3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE               3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE               3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE               3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE               3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE               3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...

选择了 9 个记录。
```

此查询（已继续）的输出。

```
... REORG_STATUS REORG_TBSPC_ID DBPARTITIONNUM DATA_PARTITION_ID
    ------------------------- -------------- -----------------
... COMPLETED                3              2                 0
... COMPLETED                3              2                 1
... COMPLETED                3              2                 2
... COMPLETED                3              1                 0
... COMPLETED                3              1                 1
... COMPLETED                3              1                 2
... COMPLETED                3              0                 0
... COMPLETED                3              0                 1
... COMPLETED                3              0                 2
```

相关概念:

- 『数据分区』（*《管理指南: 计划》*）
- 『分区表』（*《管理指南: 计划》*）
- 第 19 页的『快照监视器』

相关参考:

- 第 181 页的『data_partition_id － 数据分区标识监视元素』

- 『GET SNAPSHOT command』（*Command Reference*）
- 第 347 页的『reorg_max_phase － 最大重组阶段』
- 第 346 页的『reorg_phase_start － 重组阶段开始时间』
- 第 93 页的『快照监视器接口至逻辑数据组的映射』
- 第 96 页的『快照监视器逻辑数据组和监视元素』
- 第 45 页的『快照监视器样本输出』
- 『SNAPTAB administrative view and SNAP_GET_TAB_V91 table function – Retrieve table logical data group snapshot information』（*Administrative SQL Routines and Views*）
- 『SNAPTAB_REORG administrative view and SNAP_GET_TAB_REORG table function – Retrieve table reorganization snapshot information』（*Administrative SQL Routines and Views*）

快照监视器 SQL 管理视图

提供了许多不同的快照监视器 SQL 管理视图，每个管理视图都返回有关特定数据库系统区域的监视器数据。例如，SYSIBMADM.SNAPBP SQL 管理视图捕获快照或缓冲池信息。下表列示了每个可用的快照监视器管理视图:

表 4. 快照监视器 SQL 管理视图

监视器级别	SQL 管理视图	返回的信息
数据库管理器	SYSIBMADM.SNAPDBM	数据库管理器级别信息。
数据库管理器	SYSIBMADM.SNAPFCM	关于快速通信管理器（FCM）的数据库管理器级别信息。
数据库管理器	SYSIBMADM.SNAPFCM_PART	某个分区的关于快速通信管理器（FCM）的数据库管理器级别信息。
数据库管理器	SYSIBMADM.SNAPSWITCHES	数据库管理器监视开关设置。
数据库管理器	SYSIBMADM.SNAPDBM_MEMORY_POOL	有关内存使用情况的数据库管理器级别信息。
数据库	SYSIBMADM.SNAPDB	数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SYSIBMADM.SNAPDB_MEMORY_POOL	仅与 UNIX 平台的内存使用情况有关的数据库级别信息。
数据库	SYSIBMADM.SNAPHADR	有关高可用性灾难恢复的数据库级别信息。
应用程序	SYSIBMADM.SNAPAPPL	有关连接至数据库的每个应用程序的常规应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SYSIBMADM.SNAPAPPL_INFO	有关连接至数据库的每个应用程序的常规应用程序级别标识信息。
应用程序	SYSIBMADM.SNAPLOCKWAIT	有关连接至数据库的应用程序的锁定等待数的应用程序级别信息。
应用程序	SYSIBMADM.SNAPSTMT	有关连接至数据库的应用程序的语句的应用程序级别信息。此信息包括最近执行的 SQL 语句（如果设置了语句开关）。

表 4. 快照监视器 SQL 管理视图 (续)

监视器级别	SQL 管理视图	返回的信息
应用程序	SYSIBMADM.SNAPAGENT	有关连接至数据库的应用程序的关联代理程序的应用程序级别信息。
应用程序	SYSIBMADM.SNAPSUBSECTION	有关连接至数据库的应用程序的访问方案子节的应用程序级别信息。
应用程序	SYSIBMADM.SNAPAGENT_MEMORY_POOL	有关代理程序级别的内存使用情况的信息。
表	SYSIBMADM.SNAPTAB	每个与数据库相连接的应用程序的数据库级别和应用程序级别表活动信息。与数据库相连接的应用程序**已访问的**每个表在表级别的活动信息。需要表开关。
表	SYSIBMADM.SNAPTAB_REORG	正在重组的数据库中每个表在表级别的重组信息。
锁	SYSIBMADM.SNAPLOCK	每个与数据库相连接的应用程序的数据库级别和应用程序级别锁定信息。需要锁定开关。
表空间	SYSIBMADM.SNAPTBSP	有关以下级别的表空间活动信息：数据库级别、与数据库相连接的每个应用程序的应用程序级别以及与数据库相连接的应用程序已访问的每个表空间的表空间级别。需要缓冲池开关。
表空间	SYSIBMADM.SNAPTBSP_PART	表空间配置信息。
表空间	SYSIBMADM.SNAPTBSP_QUIESCER	表空间级别停顿者信息。
表空间	SYSIBMADM.SNAPCONTAINER	表空间级别表空间容器配置信息。
表空间	SYSIBMADM.SNAPTBSP_RANGE	表空间映射的范围信息。
缓冲池	SYSIBMADM.SNAPBP	指定数据库的缓冲池活动计数器。需要缓冲池开关。
缓冲池	SYSIBMADM.SNAPBP_PART	有关针对每个分区计算出来的缓冲区大小和使用情况的信息。
动态 SQL	SYSIBMADM.SNAPDYN_SQL	数据库的 SQL 语句高速缓存中的时间点语句信息。
数据库	SYSIBMADM.SNAPUTIL	关于实用程序的信息。
数据库	SYSIBMADM.SNAPUTIL_PROGRESS	关于实用程序进度的信息。
数据库	SYSIBMADM.SNAPDETAILLOG	有关日志文件的数据库级别信息。
数据库	SYSIBMADM.SNAPSTORAGE_PATHS	返回数据库的自动存储器路径列表，包括每个存储器路径的文件系统信息。

在捕获快照前，请考虑是否需要由监视开关控制的监视元素提供的信息。如果特定监视开关处于关闭状态，就不会收集它控制的监视元素。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

所有快照监视管理视图和关联表函数都使用单独的实例连接，该连接与当前会话使用的连接不同。因此，只有缺省数据库管理器监视开关起作用。不起作用的监视开关包括任何在当前会话或应用程序中动态打开或关闭的开关。

DB2 版本 9.1 还提供了一组管理视图，这些管理视图不仅返回各个监视元素的值，还返回监视任务通常需要的计算值。例如，SYSIBMADM.BP_HITRATIO 管理视图返回缓冲池命中率计算值，此值由各个监视元素的值累计得出的。

表 5. 快照监视器 SQL 管理公用视图

SQL 管理公用视图	返回的信息
SYSIBMADM.APPLICATIONS	关于已连接的数据库应用程序的信息。
SYSIBMADM.APPL_PERFORMANCE	有关选择的行数与应用程序已读取的行数的比率的信息。
SYSIBMADM.BP_HITRATIO	数据库中的缓冲池命中率，包括命中率总计、数据命中率和索引命中率。
SYSIBMADM.BP_READ_IO	关于缓冲池读性能的信息。
SYSIBMADM.BP_WRITE_IO	关于缓冲池写性能的信息。
SYSIBMADM.CONTAINER_UTILIZATION	关于表空间容器和利用率的信息。
SYSIBMADM.LOCKS_HELD	关于当前挂起的锁定的信息。
ISYSIBMADM.LOCKWAIT	有关代表等待获取锁定的应用程序工作的 DB2 代理程序的信息。
SYSIBMADM.LOG_UTILIZATION	有关当前连接的数据库的日志利用率的信息。
SYSIBMADM.LONG_RUNNING_SQL	有关在当前连接的数据库中花费最长时间运行的 SQL 的信息。
SYSIBMADM.QUERY_PREP_COST	关于准备不同 SQL 语句所需时间的信息。
SYSIBMADM.TBSP_UTILIZATION	表空间配置和利用率信息。
SYSIBMADM.TOP_DYNAMIC_SQL	可按执行次数、平均执行时间、排序数或每个语句的排序数进行排序的顶级动态 SQL 语句数。

相关概念:

- 第 19 页的『快照监视器』

相关任务:

- 第 52 页的『对数据库系统快照的 SQL 访问』

相关参考:

- 第 93 页的『快照监视器接口至逻辑数据组的映射』

示例: 使用快照管理视图来标识高成本应用程序

ShopMart 数据库上最近的工作负载增长已开始影响到整体数据库性能。Jessie 是 ShopMart 的 DBA，她尝试使用下列管理视图来标识日常工作负载中较大的资源使用者:

APPLICATION_PERFORMANCE

此视图帮助 Jessie 标识可能正在执行大型表扫描操作的应用程序:

```
connect to shopmart;
select AGENT_ID, ROWS_SELECTED, ROWS_READ from APPLICATION_PERFORMANCE;
```

ROWS_SELECTED 值显示返回给应用程序的行数，ROWS_READ 值显示在基本表中访问的行数。如果选择率很低，则表示该应用程序可能正在执行表扫描操作，通过创建索引可以避免应用程序执行该操作。Jessie 使用此视图来标识可能有问题的查询，然后，她可以进行进一步调查，即查看 SQL 以确定是否能够减少查询在执行时读取的行数。

LONG_RUNNING_SQL

Jessie 使用 LONG_RUNNING_SQL 管理视图来标识当前正在执行的运行时间最长的查询:

```
connect to shopmart;
select ELAPSED_TIME_MIN, APPL_STATUS, AGENT_ID from long_running_sql
order by ELAPSED_TIME_MIN desc fetch first 5 rows only;
```

通过使用此视图，她可以确定这些查询已运行的时间长度以及这些查询的状态。如果某个查询已执行了很长时间并且正在等待锁，她就可以使用对特定代理程序标识执行查询的 LOCKWAITS 或 LOCK_HELD 管理视图来进行进一步调查。LONG_RUNNING_SQL 视图还会指出正在执行的语句并允许她标识可能有问题的 SQL。

QUERY_PREP_COST

Jessie 使用 QUERY_PREP_COST 来对已确定有问题的查询进行故障诊断。此视图可以指出查询的运行频率以及这些查询中每个查询的平均执行时间:

```
connect to shopmart;
select NUM_EXECUTIONS, AVERAGE_EXECUTION_TIME_S, PREP_TIME_PERCENT from
QUERY_PREP_COST order by NUM_EXECUTIONS desc;
```

PREP_TIME_PERCENT 值向 Jessie 指出准备查询时耗用的时间在查询执行时间中所占的百分比。如果编译和优化查询时耗用的时间几乎与查询的执行时间一样长，那么，Jessie 可以建议该查询的所有者更改用于该查询的优化类。降低优化类可以使该查询更快地完成优化，从而更快地返回结果。但是，如果某个查询需要相当长的时间来进行准备，但要执行数千次（而不必再次进行准备），则更改优化类并不能提高查询性能。

TOP_DYNAMIC_SQL

Jessie 使用 TOP_DYNAMIC_SQL 视图来标识执行频率最高、运行时间最长和排序次数最多的动态 SQL 语句。有了此信息，Jessie 在进行 SQL 调整工作时就可以把注意力放在代表某些最大资源使用者的查询上。

为了标识运行频率最高的动态 SQL 语句，Jessie 发出以下语句:

```
connect to shopmart;
select * from TOP_DYNAMIC_SQL order by NUM_EXECUTIONS
desc fetch first 5 rows only;
```

此语句返回执行频率最高的 5 个动态 SQL 语句的所有执行时间、排序执行次数和语句文本详细信息。

为了标识执行时间最长的动态 SQL 语句，Jessie 检查 AVERAGE_EXECUTION_TIME_S 值最大的 5 个查询:

```
connect to shopmart;
select * from TOP_DYNAMIC_SQL order by AVERAGE_EXECUTION_TIME_S
desc fetch first 5 rows only;
```

为了查看排序次数最多的动态 SQL 语句的详细信息，Jessie 发出以下语句:

```
connect to shopmart;
select STMT_SORTS, SORTS_PER_EXECUTION, substr(STMT_TEXT,1,60) as STMT_TEXT
from TOP_DYNAMIC_SQL order by STMT_SORTS desc fetch first 5 rows only;
```

相关概念:

- 第 38 页的『示例：使用管理视图来监视缓冲池效率』

相关参考:

- 『APPL_PERFORMANCE administrative view – Retrieve percentage of rows selected for an application』(*Administrative SQL Routines and Views*)
- 『APPLICATIONS administrative view – Retrieve connected database application information』(*Administrative SQL Routines and Views*)
- 『LONG_RUNNING_SQL administrative view』(*Administrative SQL Routines and Views*)
- 『QUERY_PREP_COST administrative view – Retrieve statement prepare time information』(*Administrative SQL Routines and Views*)
- 第 34 页的『快照监视器 SQL 管理视图』
- 『TOP_DYNAMIC_SQL administrative view – Retrieve information on the top dynamic SQL statements』(*Administrative SQL Routines and Views*)

示例: 使用管理视图来监视缓冲池效率

John 是一位 DBA，他怀疑 SALES 数据库中应用程序性能不佳的原因是缓冲池的工作效率不高。为了进行调查，他使用 BP_HITRATIO 管理视图来查看缓冲池命中率:

```
connect to SALES;
select BPNAME, TOTAL_HIT_RATIO from BP_HIT_RATIO;
```

John 看到其中一个缓冲池的命中率非常低，这表示从磁盘读取了太多的页（而不是从缓冲池读取那些页）。

然后，他决定使用 BP_READ_IO 管理视图来了解是否需要对预取程序进行调整:

```
connect to SALES;
select BPNAME, PERCENT_SYNC_READS,UNUSED_ASYNC_READS_PERCENT from BP_READ_IO;
```

PERCENT_SYNC_READS 值指示了在未进行预取的情况下以异步方式读取的页所占的百分比。如果数值比较大，则表示有很大一部分数据是直接从磁盘读取的，这意味着需要更多的预取程序。UNUSED_ASYNC_READS_PERCENT 值指示以异步方式从磁盘读取但查询从未访问过的页所占的百分比。如果此数值较大，则表示预取程序过度地读取数据页，从而产生不必要的 I/O。

由于 PERCENT_SYNC_READS 和 UNUSED_ASYNC_READS_PERCENT 值似乎都在可接受的范围内，所以 John 使用 BP_WRITE_IO 管理视图来调查页清除程序为传入数据页清除空间的效率:

```
connect to SALES;
select BPNAME, PERCENT_WRITES_ASYNC from BP_WRITE_IO;
```

PERCENT_WRITES_ASYNC 值指示以异步方式执行的物理写请求所占的百分比。如果此数值较大，则表示页清除程序能够很好地在传入新数据页请求之前清除缓冲池空间。如果此数值较小，则表示数据库代理进程执行了大量的物理写操作（在此期间，应用程序将等待数据页被读入缓冲池）。

John 看到 PERCENT_WRITES_ASYNC 值为 25%，这是一个非常小的值，因此他决定为 SALES 数据库配置更多页清除程序以提高异步写比率。在增加页清除程序数之后，他可以再次使用缓冲池管理视图来查看调整效果。

相关概念:

- 第 36 页的『示例: 使用快照管理视图来标识高成本应用程序』

相关参考:

- 『BP_HITRATIO administrative view – Retrieve bufferpool hit ratio information』（*Administrative SQL Routines and Views*）
- 『BP_READ_IO administrative view – Retrieve bufferpool read performance information』（*Administrative SQL Routines and Views*）
- 『BP_WRITE_IO administrative view – Retrieve bufferpool write performance information』（*Administrative SQL Routines and Views*）
- 第 34 页的『快照监视器 SQL 管理视图』

从 CLP 捕获数据库快照

可使用 GET SNAPSHOT 命令从 CLP 捕获数据库快照。有若干不同快照请求类型可用，可通过对 GET SNAPSHOT 命令指定特定参数来访问它们。

必须具有实例连接才能捕获数据库快照。如果没有实例连接，则创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

先决条件:

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能捕获数据库快照。

过程:

1. 可选：设置并检查监视开关的状态。
2. 从 CLP 发出带有期望参数的 `GET SNAPSHOT` 命令。在以下示例中，快照将捕获数据库管理器级别信息:
   ```
   db2 get snapshot for dbm
   ```
3. 对于分区数据库系统，可为特定分区捕获专门的数据库快照，或者为所有分区捕获全局的数据库快照。要对特定分区（如分区号 2）上的所有应用程序捕获数据库快照，请发出以下命令:
   ```
   db2 get snapshot for all applications at dbpartitionnum 2
   ```
   要对所有分区上的所有应用程序捕获数据库快照，请发出以下命令:
   ```
   db2 get snapshot for all applications global
   ```
   对于分区数据库上的全局快照，将聚集所有分区中的监视器数据。

相关概念:

- 第 48 页的『分区数据库系统上的全局快照』
- 第 19 页的『快照监视器』
- 第 11 页的『系统监视开关』

相关任务:

- 第 12 页的『通过 CLP 设置监视开关』

相关参考:

- 『GET SNAPSHOT command』（*Command Reference*）
- 第 40 页的『快照监视器 CLP 命令』
- 第 93 页的『快照监视器接口至逻辑数据组的映射』

- 第 45 页的『快照监视器样本输出』

快照监视器 CLP 命令

下表列示了所有受支持的快照请求类型。对于特定请求类型，仅当相关联的监视开关设置为 ON 时，才返回某些信息。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

表 6. 快照监视器 CLP 命令

监视器级别	CLP 命令	返回的信息
连接列表	list applications [show detail]	当前连接至数据库的所有应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
连接列表	list applications for database *dbname* [show detail]	当前连接至指定数据库的每个应用程序的标识信息。
连接列表	list dcs applications	当前连接至数据库的所有 DCS 应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
数据库管理器	get snapshot for dbm	数据库管理器级别信息，包括实例级别监视开关设置。
数据库管理器	get dbm monitor switches	实例级别监视开关设置。
数据库	get snapshot for database on *dbname*	数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	get snapshot for all databases	在分区上处于活动状态的每个数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	list active databases	与每个活动数据库的连接的数目。包括使用 ACTIVATE DATABASE 命令启动但没有连接的数据库。
数据库	get snapshot for dcs database on *dbname*	特定 DCS 数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	get snapshot for remote database on *dbname*	特定联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	get snapshot for all remote databases	分区上的每个活动联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
应用程序	get snapshot for application applid *appl-id*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	get snapshot for application agentid *appl-handle*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	get snapshot for applications on *dbname*	与分区中数据库相连接的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	get snapshot for all applications	在分区中处于活动状态的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	get snapshot for dcs application applid *appl-id*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	get snapshot for all dcs applications	在分区中处于活动状态的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	get snapshot for dcs application agentid *appl-handle*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	get snapshot for dcs applications on *dbname*	与分区中数据库相连接的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。

表 6. 快照监视器 CLP 命令 (续)

监视器级别	CLP 命令	返回的信息
应用程序	get snapshot for remote applications on *dbname*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	get snapshot for all remote applications	在分区中处于活动状态的每个联合系统应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
表	get snapshot for tables on *dbname*	每个与数据库相连接的应用程序的数据库级别和应用程序级别表活动信息。与数据库相连接的应用程序**已访问的**每个表在表级别的活动信息。需要表开关。
锁	get snapshot for locks for application applid *appl-id*	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁	get snapshot for locks for application agentid *appl-handle*	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁	get snapshot for locks on *dbname*	每个与数据库相连接的应用程序的数据库级别和应用程序级别锁定信息。需要锁定开关。
表空间	get snapshot for tablespaces on *dbname*	有关数据库的表空间活动的信息。需要缓冲池开关。还包括有关容器、停顿者和范围的信息。此信息不受开关控制。
缓冲池	get snapshot for all bufferpools	缓冲池活动计数器。需要缓冲池开关。
缓冲池	get snapshot for bufferpools on *dbname*	指定数据库的缓冲池活动计数器。需要缓冲池开关。
动态 SQL	get snapshot for dynamic sql on *dbname*	数据库的 SQL 语句高速缓存中的时间点语句信息。该信息也可能来自远程数据源。

相关任务:

- 第 39 页的『从 CLP 捕获数据库快照』
- 第 12 页的『通过 CLP 设置监视开关』

相关参考:

- 『GET SNAPSHOT command』（*Command Reference*）
- 第 93 页的『快照监视器接口至逻辑数据组的映射』

从客户机应用程序捕获数据库快照

可使用 C、C++ 或 COBOL 应用程序中的快照监视器 API 来捕获数据库快照。在 C 和 C++ 中，可通过在 db2GetSnapshot() 参数中指定特定参数来访问若干不同快照请求类型。

必须具有实例连接才能捕获数据库快照。如果没有实例连接，则创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

先决条件:

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能使用 db2MonitorSwitches API。

过程:

1. 可选：设置并检查监视开关的状态。

2. 包括下列 DB2 库：sqlmon.h 和 db2ApiDf.h。可在 `sqllib` 中的 `include` 子目录下找到它们。

```
#include <db2ApiDf.h>
#include <sqlmon.h>
```

3. 将快照缓冲区单元大小设置为 100 KB。

```
#define SNAPSHOT_BUFFER_UNIT_SZ 102400
```

4. 声明 sqlca、sqlma、db2GetSnapshotData 和 sqlm_collected 结构。还应初始化指针以包含快照缓冲区并确定缓冲区大小。

```
struct sqlma *pRequestedDataGroups;
struct sqlca sqlca;
memset (&sqlca, '\0', sizeof(struct sqlca));
struct sqlm_collected collectedData;
memset (&collectedData, '\0', sizeof(collectedData));
db2GetSnapshotData getSnapshotParam;
memset (&getSnapshotParam, '\0', sizeof(getSnapshotParam));

static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
sqluint32 outputFormat;
char *snapshotBuffer;
```

5. 初始化 sqlma 结构，并指定要捕获的快照属于数据库管理器级别信息。

```
pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
memset(pRequestedDataGroups, '\0', SQLMASIZE(1));
pRequestedDataGroups->obj_num = 1;
pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
```

6. 初始化缓冲区以容纳快照输出。

```
  snapshotBuffer = (char *)malloc(snapshotBufferSize);
memset (snapshotBuffer, '\0', snapshotBufferSize);
```

7. 使用快照请求类型（来自 sqlma 结构）、缓冲区信息和捕获快照所需的其他信息来填充 db2GetSnapshotData 结构。

```
getSnapshotParam.piSqlmaData = pRequestedDataGroups;
getSnapshotParam.poCollectedData = &collectedData;
getSnapshotParam.poBuffer = snapshotBuffer;
getSnapshotParam.iVersion = SQLM_DBMON_VERSION8;
getSnapshotParam.iBufferSize = snapshotBufferSize;
getSnapshotParam.iStoreResult = 0;
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_DEFAULT;
```

8. 捕获快照。传递 db2GetSnapshotData 结构和对缓冲区的引用，db2GetSnapshotData 结构包含捕获快照所需的信息，快照输出将引导至该缓冲区。

```
  db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
```

9. 包括用于处理缓冲区溢出的逻辑。获取快照后，将检查 sqlcode 是否存在缓冲区溢出。如果发生了缓冲区溢出，则将清除并重新初始化缓冲区，然后再次获取快照。

```
while (sqlca.sqlcode == 1606)
{
free(snapshotBuffer);
  snapshotBufferSize = snapshotBufferSize +
  SNAPSHOT_BUFFER_UNIT_SZ;
  snapshotBuffer = (char *)malloc(snapshotBufferSize);
  if (snapshotBuffer == NULL)
  {
    printf("\nMemory allocation error.\n");
    return 1;
  }
```

```
    getSnapshotParam.iBufferSize = snapshotBufferSize;
    getSnapshotParam.poBuffer = snapshotBuffer;
    db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
  }
```

10. 处理快照监视器数据流。
11. 清除缓冲区。

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

相关概念:

- 第 19 页的『快照监视器』
- 第 49 页的『快照监视器自描述数据流』
- 第 7 页的『系统监视器输出: 自描述数据流』
- 第 11 页的『系统监视开关』

相关任务:

- 第 14 页的『通过客户机应用程序设置监视开关』

相关参考:

- 『db2GetSnapshot API - Get a snapshot of the database manager operational status』(*Administrative API Reference*)
- 『db2GetSnapshotSize API - Estimate the output buffer size required for the db2GetSnapshot API』(*Administrative API Reference*)
- 『db2MonitorSwitches API - Get or update the monitor switch settings』(*Administrative API Reference*)
- 『db2ResetMonitor API - Reset the database system monitor data』(*Administrative API Reference*)
- 第 44 页的『快照监视器 API 请求类型』

相关样本:

- 『dbsnap.cbl -- Get a database monitor snapshot (IBM COBOL)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C)』
- 『clisnap.c -- Capture a snapshot at the client level (C)』
- 『dbsnap.c -- Capture a snapshot at the database level (C)』
- 『insnap.c -- Capture a snapshot at the instance level (C)』
- 『utilsnap.c -- Utilities for the snapshot monitor samples (C)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C++)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C++)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C++)』
- 『clisnap.C -- Capture a snapshot at the client level (C++)』
- 『dbsnap.C -- Capture a snapshot at the database level (C++)』
- 『insnap.C -- Capture a snapshot at the instance level (C++)』
- 『utilsnap.C -- Utilities for the snapshot monitor samples (C++)』

快照监视器 API 请求类型

下表列示了所有受支持的快照请求类型。对于特定请求类型，仅当相关联的监视开关设置为 ON 时，才返回某些信息。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

表 7. 快照监视器 API 请求类型

监视器级别	API 请求类型	返回的信息
连接列表	SQLMA_APPLINFO_ALL	当前连接至数据库的所有应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
连接列表	SQLMA_DBASE_APPLINFO	当前连接至指定数据库的每个应用程序的标识信息。
连接列表	SQLMA_DCS_APPLINFO_ALL	当前连接至数据库的所有 DCS 应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
数据库管理器	SQLMA_DB2	数据库管理器级别信息，包括实例级别监视开关设置。
数据库	SQLMA_DBASE	数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE_ALL	在分区上处于活动状态的每个数据库的数据库级别信息和计数器。与每个活动数据库的连接的数目。包括使用 ACTIVATE DATABASE 命令启动但没有连接的数据库。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DCS_DBASE	特定 DCS 数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DCS_DBASE_ALL	在分区上处于活动状态的每个 DCS 数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE_REMOTE	特定联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE_REMOTE_ALL	分区上的每个活动联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
应用程序	SQLMA_APPL	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_AGENT_ID	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DBASE_APPLS	与分区中数据库相连接的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_APPL_ALL	在分区中处于活动状态的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_APPL	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_APPL_ALL	在分区中处于活动状态的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_APPL_HANDLE	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_DBASE_APPLS	与分区中数据库相连接的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。

表 7. 快照监视器 API 请求类型 (续)

监视器级别	API 请求类型	返回的信息
应用程序	SQLMA_DBASE_APPLS_REMOTE	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_APPL_REMOTE_ALL	在分区中处于活动状态的每个联合系统应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
表	SQLMA_DBASE_TABLES	每个与数据库相连接的应用程序的数据库级别和应用程序级别表活动信息。与数据库相连接的应用程序**已访问的**每个表在表级别的活动信息。需要表开关。
锁	SQLMA_APPL_LOCKS	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁	SQLMA_APPL_LOCKS_AGENT_ID	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁	SQLMA_DBASE_LOCKS	每个与数据库相连接的应用程序的数据库级别和应用程序级别锁定信息。需要锁定开关。
表空间	SQLMA_DBASE_TABLESPACES	有关以下级别的表空间活动信息：数据库级别、与数据库相连接的每个应用程序的应用程序级别以及与数据库相连接的应用程序已访问的每个表空间的表空间级别。需要缓冲池开关。
缓冲池	SQLMA_BUFFERPOOLS_ALL	缓冲池活动计数器。需要缓冲池开关。
缓冲池	SQLMA_DBASE_BUFFERPOOLS	指定数据库的缓冲池活动计数器。需要缓冲池开关。
动态 SQL	SQLMA_DYNAMIC_SQL	数据库的 SQL 语句高速缓存中的时间点语句信息。

相关概念:

- 第 19 页的『快照监视器』
- 第 49 页的『快照监视器自描述数据流』

相关任务:

- 第 41 页的『从客户机应用程序捕获数据库快照』
- 第 14 页的『通过客户机应用程序设置监视开关』

相关参考:

- 『db2ConvMonStream API - Convert the monitor stream to the pre-version 6 format』（*Administrative API Reference*）
- 『db2GetSnapshot API - Get a snapshot of the database manager operational status』（*Administrative API Reference*）
- 『db2GetSnapshotSize API - Estimate the output buffer size required for the db2GetSnapshot API』（*Administrative API Reference*）
- 『db2MonitorSwitches API - Get or update the monitor switch settings』（*Administrative API Reference*）
- 『db2ResetMonitor API - Reset the database system monitor data』（*Administrative API Reference*）

快照监视器样本输出

为说明快照监视器的特征，下面提供了使用 CLP 获取快照的示例及相应输出。此示例的目的在于获取连接至 SAMPLE 数据库的应用程序挂起的锁定列表。采取的步骤如下所示:

1. 连接至样本数据库:

```
db2 connect to sample
```

2. 使用 UPDATE MONITOR SWITCHES 命令将 LOCK 开关设置为 ON，这样就会收集等待锁定所花的时间：

```
db2 update monitor switches using LOCK on
```

3. 发出需要针对数据库目录的锁定的命令或语句。在此情况下，我们将声明、打开和访存游标：

```
db2 -c- declare c1 cursor for
                       select * from staff where job='Sales' for update
db2 -c- open c1
db2 -c- fetch c1
```

4. 使用 GET SNAPSHOT 命令获取数据库锁定快照：

```
db2 get snapshot for locks on sample
```

在从 CLP 发出 GET SNAPSHOT 命令后，快照输出将引导至屏幕。

```
            数据库锁定快照

数据库名称                              = SAMPLE
数据库路径                              = C:\DB2\NODE0000\SQL00001\
输入数据库别名                          = SAMPLE
     挂起的锁定数                       = 5
     当前连接的应用程序数               = 1
当前正在等待锁定的代理程序数            = 0
快照时间戳记                            = 06-05-2002 17:08:25.048027

应用程序句柄                            = 8
应用程序标识                            = *LOCAL.DB2.0098C5210749
序号                                    = 0001
应用程序名称                            = db2bp.exe
连接授权标识                            = DB2ADMIN
应用程序状态                            = UOW 正在等待
状态更改时间                            = 未收集
应用程序代码页                          = 1252
     挂起的锁定数                       = 5
     总等待时间（ms）                   = 0

锁定列表
 锁定名称                 = 0x02000300050000000000000052
 锁定属性                 = 0x00000000
 释放标志                 = 0x00000001
 锁定计数                 = 1
 挂起计数                 = 0
 锁定对象名               = 5
 对象类型                 = 行
 表空间名称               = USERSPACE1
 表模式                   = DB2ADMIN
 表名                     = STAFF
 方式                     = U

 锁定名称                 = 0x02000300000000000000000054
 锁定属性                 = 0x00000000
 释放标志                 = 0x00000001
 锁定计数                 = 1
 挂起计数                 = 0
 锁定对象名               = 3
 对象类型                 = 表
 表空间名称               = USERSPACE1
 表模式                   = DB2ADMIN
 表名                     = STAFF
 方式                     = IX

 锁定名称                 = 0x01000000010000000100810056
 锁定属性                 = 0x00000000
```

```
释放标识                   = 0x40000000
锁定计数                   = 1
挂起计数                   = 0
锁定对象名                 = 0
对象类型                   = 内部偏差锁定
方式                       = S

锁定名称                   = 0x41414141414A48520000000041
锁定属性                   = 0x00000000
释放标识                   = 0x40000000
锁定计数                   = 1
挂起计数                   = 0
锁定对象名                 = 0
对象类型                   = 内部计划锁定
方式                       = S

锁定名称                   = 0x434F4E544F4B4E310000000041
锁定属性                   = 0x00000000
释放标识                   = 0x40000000
锁定计数                   = 1
挂起计数                   = 0
锁定对象名                 = 0
对象类型                   = 内部计划锁定
方式                       = S
```

可从此快照中看到，当前有一个应用程序连接至 SAMPLE 数据库，并且该应用程序挂起 5 个锁定。

```
挂起的锁定数                                  = 5
当前连接的应用程序数                          = 1
```

注意，应用程序状态成为 UOW 正在锁定的时间（状态更改时间）返回为未收集。这是因为 UOW 开关为 OFF。

锁定快照还将返回直到目前为止连接至此数据库的应用程序等待锁定所花的总时间。

```
总等待时间（ms）                   = 0
```

相关概念:

- 第 19 页的『快照监视器』
- 第 11 页的『系统监视开关』

相关任务:

- 第 12 页的『通过 CLP 设置监视开关』

相关参考:

- 第 40 页的『快照监视器 CLP 命令』

子节快照

在使用分区间并行性的系统上，SQL 编译器将 SQL 语句的访问方案分为若干子节。每个子节由一个不同的 DB2 代理程序（或 SMP 的代理程序）执行。

通过使用 db2expln 或 dynexpln 命令，可获取 DB2 代码生成器在编译期间生成的 SQL 语句的访问方案。例如，选择分布在若干分区上的表中的所有行可能导致访问方案有两个子节:

1. 子节 0，协调程序子节，作用是收集其他 DB2 代理程序访存的行并将其返回至应用程序。

2. 子节 1，作用是执行表扫描并将行返回至协调代理程序。

在此简单示例中，子节 1 将分布在所有数据库分区上。数据库分区组的每个物理分区上都有执行此子节的子代理程序，此表属于该数据库分区组。

数据库系统监视器允许您将运行时信息与访问方案相关联，这是编译时信息。借助分区间并行性，监视器会中断信息并下行至子节级别。例如，当语句监视开关设置为 ON 时，GET SNAPSHOT FOR APPLICATION 将返回在此分区上执行的每个子节的信息以及语句总数。

对应用程序快照返回的子节信息包括:
- 读取／写入的表行数
- CPU 消耗
- 耗用时间
- 发送至其他代理程序和从其他代理程序接收的表队列行数，这些代理程序处理此语句。这允许您通过获取一系列快照来跟踪长时间运行的查询的执行情况。
- 子节状态。如果子节因为等待另一代理程序发送或接收数据而处于 WAIT 状态，则该信息还会标识阻止子节继续执行的分区。然后可获取这些分区的快照来调查情况。

在每个子节执行完之后，语句事件监视器记录的有关该子节的信息包括：CPU 消耗、总执行时间和若干其他计数器。

相关概念:
- 第 48 页的『分区数据库系统上的全局快照』
- 第 19 页的『快照监视器』
- 第 11 页的『系统监视开关』

相关参考:
- 第 40 页的『快照监视器 CLP 命令』
- 『GET SNAPSHOT command』（*Command Reference*）

分区数据库系统上的全局快照

在分区数据库系统上，可获取当前分区、指定分区或所有分区的快照。对分区数据库的所有分区获取全局快照时，会先聚集数据，然后返回结果。

对不同元素类型聚集数据的方式如下所示:
- **计数器、时间和标尺**

  包含从实例中的每个分区收集的所有可能值的总和。例如，GET SNAPSHOT FOR DATABASE XYZ ON TEST GLOBAL 对分区数据库实例中的所有分区返回从数据库读取的行数（rows_read）。
- **水位标记**

  返回分区数据库系统中的任何分区的最高（高水位）或最低（低水位）值。如果返回的值值得关注，则可以获取各个分区的快照以确定特定分区是否使用过度或者问题是否为实例范围内的问题。
- **时间戳记**

设置为连接快照监视器实例代理程序的分区的时间戳记值。注意，所有时间戳记值都在 `timestamp` 监视开关控制之下。

- **信息**

  返回可能妨碍工作的分区的最重要信息。例如，对于元素 `appl_status`，如果一个分区上的状态为“正在执行 UOW”，而另一个分区上的状态为“等待锁定”，则返回“等待锁定”，原因是这是挂起应用程序的执行的状态。

您也可以复位计数器，设置监视开关，以及检索分区数据库中的个别分区或所有分区的监视开关设置。

注: 获取全局快照时，如果一个或多个分区遇到错误，则将从成功获取快照的分区收集数据，同时返回一个警告（sqlcode 1629）。如果以全局方式获取或更新监视开关，或者计数器在一个或多个分区上复位失败，则不会设置这些分区的监视开关或进行数据复位。

相关概念:

- 第 6 页的『计数器状态和可视性』
- 第 19 页的『快照监视器』
- 第 47 页的『子节快照』

相关任务:

- 第 41 页的『从客户机应用程序捕获数据库快照』
- 第 39 页的『从 CLP 捕获数据库快照』
- 第 52 页的『对数据库系统快照的 SQL 访问』

快照监视器自描述数据流

在使用 db2GetSnapshot API 捕获快照之后，API 以自描述数据流的形式返回快照输出。第 50 页的图 2 显示数据流的结构，而第 51 页的表 8 提供可能返回的逻辑数据组和监视元素的一些样本。

注: 描述性名称用于示例和表中的标识。在实际数据流中，将在这些名称前加上 **SQLM_ELM_** 前缀。例如，COLLECTED 在快照监视器输出中将显示为 SQLM_ELM_COLLECTED。在实际数据流中，将在类型前加上 **SQLM_TYPE_** 前缀。例如，HEADER 在数据流中将显示为 SQLM_TYPE_HEADER。

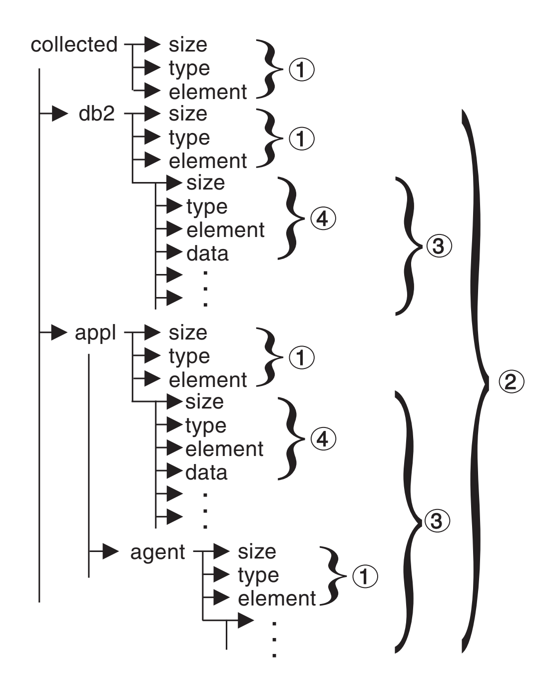


图 2. 快照监视器数据流

1. 每个逻辑数据组以指示其大小和名称的头开始。此大小不包括头本身占用的数据量。
2. 已收集头中的大小返回快照的总大小。
3. 其他头中的大小元素指示该逻辑数据组中的所有数据的大小，包括所有下级分组。
4. 监视元素信息在逻辑数据组头后面，并且也是自描述格式。

表 8. 样本快照数据流

逻辑数据组	数据流	描述
collected	1000 header collected	快照数据的大小（以字节计）。 指示逻辑数据组的开头。 逻辑数据组的名称。
	4 u32bit server_db2_type sqlf_nt_server	此监视元素中存储的数据的大小。 监视元素类型 － 不带符号的 32 位数字。 收集的监视元素的名称。 此元素的已收集值。
	2 u16bit node_number 3	此监视元素中存储的数据的大小。 监视元素类型 － 不带符号的 16 位数字。 收集的监视元素的名称。 此元素的已收集值。
db2	200 header db2	快照中的 DB2 级别数据部分的大小。 指示逻辑数据组的开头。 逻辑数据组的名称。
	4 u32bit sort_heap_allocated 16	此监视元素中存储的数据的大小。 监视元素类型 － 不带符号的 32 位数字。 收集的监视元素的名称。 此元素的已收集值。
	4 u32bit local_cons 3	此监视元素中存储的数据的大小。 监视元素类型 － 不带符号的 32 位数字。 收集的监视元素的名称。 此元素的已收集值。
	. . .	. . .
appl	100 header appl	快照中的应用程序元素数据的大小。 指示逻辑数据组的开头。 逻辑数据组的名称。
	4 u32bit locks_held 3	此监视元素中存储的数据的大小。 监视元素类型 － 不带符号的 32 位数字。 收集的监视元素的名称。 此元素的已收集值。
	. . .	. . .
agent	50 header agent	应用程序结构的代理程序部分的大小。 指示逻辑数据组的开头。 逻辑数据组的名称。
	4 u32bit agent_pid 12	此监视元素中存储的数据的大小。 监视元素类型 － 32 位数字。 收集的监视元素的名称。 此元素的已收集值。
	. . .	. . .

db2GetSnapshot() 例程在用户提供的缓冲区中返回自描述格式快照数据。将在与要捕获的快照类型相关联的逻辑数据分组中返回数据。

快照请求返回的每一项都包含指定大小和类型的字段。该字段可用来对返回的数据进行语法分析。字段的大小还可用来跳过逻辑数据组。例如，要跳过 DB2 记录，需要确定数据流中的字节数。使用以下公式来计算要跳过的字节数:

```
DB2 逻辑数据分组的大小 + sizeof(sqlm_header_info)
```

相关概念:

- 第 19 页的『快照监视器』

相关任务:

- 第 41 页的『从客户机应用程序捕获数据库快照』

相关参考:

- 第 44 页的『快照监视器 API 请求类型』
- 第 93 页的『快照监视器接口至逻辑数据组的映射』

相关样本:

- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C)』
- 『clisnap.c -- Capture a snapshot at the client level (C)』
- 『dbsnap.c -- Capture a snapshot at the database level (C)』
- 『insnap.c -- Capture a snapshot at the instance level (C)』
- 『utilsnap.c -- Utilities for the snapshot monitor samples (C)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C++)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C++)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C++)』
- 『clisnap.C -- Capture a snapshot at the client level (C++)』
- 『dbsnap.C -- Capture a snapshot at the database level (C++)』
- 『insnap.C -- Capture a snapshot at the instance level (C++)』
- 『utilsnap.C -- Utilities for the snapshot monitor samples (C++)』

对数据库系统快照的 SQL 访问

有两种方法可用来使用快照监视器 SQL 表函数（又称为*快照表函数*）访问快照监视器数据:

- 直接访问
- 文件访问

直接访问

授权用户可使用快照表函数发出查询并接收包含监视数据的结果集。如果使用此方法，则只有具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限的用户才能访问快照监视器数据。

要使用直接访问来捕获快照信息:

1. 可选: 设置并检查监视开关的状态。
2. 使用 SQL 捕获数据库系统快照。

文件访问

授权用户调用 SNAPSHOT_FILEW 存储过程（标识快照请求类型）及受影响的分区和数据库。然后 SNAPSHOT_FILEW 存储过程会将监视器数据存储在数据库服务器上的文件中。

授权用户可对其调用 SNAPSHOT_FILEW 存储过程的每个请求类型

虽然这是允许所有用户访问快照监视器数据的安全方法，但此方法仍然有一些局限性:

- SNAPSHOT_FILEW 文件中的快照监视器数据仅仅是上次调用 SNAPSHOT_FILEW 存储过程时的最新数据。通过定期调用 SNAPSHOT_FILEW 存储过程，可以确保获得最新的快照监视器数据。例如，在 UNIX 系统上，可以设置 cron 作业来完成此操作。
- 发出包含快照表函数的查询的用户不能标识所要监视的数据库或分区。发出 SNAPSHOT_FILEW 调用的用户所标识的数据库名称和分区号确定了快照表函数可访问的文件内容。
- 如果用户发出包含快照表函数的 SQL 查询，但尚未针对该快照表函数运行相应的 SNAPSHOT_FILEW 请求类型，就会尝试对当前连接的数据库和分区创建直接快照。仅当用户具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限时，此操作才会成功。

捕获数据库系统快照信息并保存至文件的 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 用户将执行下列任务。

1. 了解将发出快照请求的用户的需要。具体地说，确定他们需要的监视器数据、要从中进行收集的数据库以及收集是否限制为特定分区。
2. 可选：设置并检查监视开关的状态。
3. 捕获数据库系统快照信息并保存至文件。

一旦 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 用户完成上述步骤，所有用户都可以使用 SQL 查询中的快照表函数来访问数据库系统快照信息。

相关概念:

- 第 19 页的『快照监视器』
- 第 11 页的『系统监视开关』

相关任务:

- 第 12 页的『通过 CLP 设置监视开关』
- 第 20 页的『使用快照管理视图和表函数来捕获数据库系统快照』
- 第 22 页的『使用 SNAP_WRITE_FILE 存储过程来将数据库系统快照信息捕获到文件中』
- 第 25 页的『使用 SQL 查询中的快照表函数来访问数据库系统快照（使用文件访问）』

相关参考:

- 第 93 页的『快照监视器接口至逻辑数据组的映射』
- 第 34 页的『快照监视器 SQL 管理视图』

第 4 章 使用事件监视器

事件监视器

在指定的事件发生时，事件监视器用来收集有关数据库和所有已连接的应用程序的信息。事件表示数据库活动（如连接、死锁、语句或事务）中的转换。可按想要监视的事件的类型来定义事件监视器。例如，死锁事件监视器等待死锁发生；发生死锁时，它将收集所涉及应用程序和处于争用状态的锁定的信息。

在缺省情况下，所有数据库都有定义为 DB2DETAILDEADLOCK 的事件监视器，它将记录有关死锁事件的详细信息。DB2DETAILDEADLOCK 事件监视器将在数据库启动时自动启动。

虽然快照监视器通常用于预防性维护和问题分析，但事件监视器将用于警告管理员立即处理问题或跟踪悬而未决的问题。

要创建事件监视器，请使用 CREATE EVENT MONITOR SQL 语句。事件监视器仅在活动时才收集事件数据。要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE SQL 语句。事件监视器的状态（活动还是不活动）可通过 SQL 函数 EVENT_MON_STATE 确定。

执行 CREATE EVENT MONITOR SQL 语句时，它创建的事件监视器定义将存储在下列数据库系统目录表中:

- SYSCAT.EVENTMONITORS: 为数据库定义的事件监视器。
- SYSCAT.EVENTS: 对数据库监视的事件。
- SYSCAT.EVENTTABLES: 表事件监视器的目标表。

每个事件监视器有自己的专用逻辑视图，用于查看监视元素中的实例数据。如果释放并重新激活特定事件监视器，则这些计数器的视图将会复位。只有新激活的事件监视器受到影响；所有其他事件监视器将继续使用它们的计数器值（及所有新的添加内容）的视图。

事件监视器输出可引导至非分区 SQL 表、文件或命名管道。

相关概念:

- 第 3 页的『数据库系统监视器』

相关任务:

- 第 57 页的『收集关于数据库系统事件的信息』
- 第 59 页的『创建事件监视器』

相关参考:

- 第 75 页的『事件监视器样本输出』
- 第 56 页的『事件类型』

事件类型

事件监视器返回有关 CREATE EVENT MONITOR 语句中指定的事件类型的信息。对于每个事件类型，将在特定时间点收集监视信息。下表列示收集监视数据时的可用事件类型和每个事件类型的可用信息。第一列中的可用事件类型对应定义事件类型的 CREATE EVENT MONITOR 语句中使用的关键字。

除了出现数据的已定义事件外，还可使用 FLUSH EVENT MONITOR SQL 语句来生成事件。此方法生成的事件将使用所有监视器类型（DEADLOCKS 和 DEADLOCKS WITH DETAILS 除外）的当前数据库监视器值写入，这些监视器类型与清空的事件监视器相关联。

表 9. 事件类型

事件类型	收集数据的时间	可用信息
DEADLOCKS	死锁检测	涉及的应用程序及处于争用状态的锁定。
DEADLOCKS WITH DETAILS	死锁检测	有关涉及的应用程序的综合信息，包括参与语句（和语句文本）的标识和要挂起的锁定的列表。如果使用 DEADLOCKS WITH DETAILS 事件监视器而不是 DEADLOCKS 事件监视器，则会导致发生死锁时性能下降，原因是收集了其他的信息。
DEADLOCKS WITH DETAILS HISTORY	死锁检测	DEADLOCKS WITH DETAILS 事件监视器中报告的所有信息以及每个应用程序的当前工作单元的语句历史记录，这些应用程序拥有的锁定参与了挂起该锁定的数据库分区的死锁方案。如果使用 DEADLOCKS WITH DETAILS HISTORY 事件监视器，则会导致激活时性能轻微下降，原因是进行了语句历史记录跟踪。
DEADLOCKS WITH DETAILS HISTORY VALUES	死锁检测	带有详细信息的死锁历史记录中报告的所有信息，以及在执行语句时对所有参数标记提供的值。如果使用 DEADLOCKS WITH DETAILS HISTORY VALUES 事件监视器，则会导致激活时性能较为严重的下降，原因是额外复制数据值。
STATEMENTS	SQL 语句结束	语句启动或停止时间、使用的 CPU、动态 SQL 的文本、SQLCA（SQL 语句的返回码）及其他度量值，如访存计数。 **注：** 当时间戳记开关设置为 OFF 时，语句启动或停止时间不可用。
	子节结束	对于分区数据库：耗用的 CPU、执行时间以及表和表队列信息。
TRANSACTIONS	工作单元结束	UOW 工作启动或停止时间、先前的 UOW 时间、耗用的 CPU 以及锁定和记录度量值。如果使用 XA 运行，则不会生成事务记录。
CONNECTIONS	连接结束	所有应用程序级别计数器。
DATABASE	数据库释放	所有数据库级别计数器。
BUFFERPOOLS	数据库释放	缓冲池、预取程序、页清除程序和每个缓冲池的直接 I/O 的计数器。
TABLESPACES	数据库释放	缓冲池、预取程序、页清除程序和每个表空间的直接 I/O 的计数器。
TABLES	数据库释放	对每个表读取或写入的行数。

注： 将为每个新创建的数据库创建详细的死锁事件监视器。此事件监视器称为 DB2DETAILDEADLOCK，将在激活数据库时启动，并且写至数据库目录中的文件。可通过删除它来避免此事件监视器导致的开销。

相关概念:

- 第 6 页的『计数器状态和可视性』
- 第 55 页的『事件监视器』
- 第 124 页的『事件类型至逻辑数据组的映射』

相关任务:

- 第 57 页的『收集关于数据库系统事件的信息』
- 第 59 页的『创建事件监视器』

相关参考:

- 『DROP statement』（*SQL Reference, Volume 2*）
- 第 75 页的『事件监视器样本输出』

收集关于数据库系统事件的信息

使用 SQL 数据定义语言（SQL DDL）语句创建并处理事件监视器和数据库对象之类的内容。下面列示的步骤表示事件监视器的典型生命周期。如果真是如此，也不一定要按显示的顺序执行这些步骤。例如，根据使用情况，事件监视器可能永远不会被删除，甚至不会被释放。

但是，事件监视器生命周期中有两项是不变的。

1. 第一步总是创建事件监视器。
2. 最后一步总是删除事件监视器。

先决条件:

您将需要 DBADM 权限来创建和操作事件监视器。

过程:

1. 创建事件监视器。创建事件监视器。
2. *仅适用于文件和管道事件监视器*: 确保将接收事件记录的目录或命名管道存在。事件监视器本来不会激活。

   在 AIX® 上，可使用 mkfifo 命令来创建命名管道。在 Linux® 和其他 UNIX 类型（如 Solaris 操作系统）上，使用 pipe() 例程。

   在 Windows® 2000 上，可通过使用 CreateNamedPipe() 例程来创建命名管道。
3. *仅适用于管道事件监视器*: 在激活事件监视器前打开命名管道。可使用操作系统功能来完成此操作:
   - 对于 UNIX: `open()`
   - 对于 Windows 2000: `ConnectNamedPipe()`

   还可使用 db2evmon 可执行文件来完成此操作:

   ```
   db2evmon -db databasename
           -evm eventmonname
   ```

   `databasename` 表示被监视的数据库的名称。

   `evmonname` 表示事件监视器的名称。
4. 激活新创建的事件监视器以使它能够收集信息。

```
SET EVENT MONITOR evmonname STATE 1;
```

如果事件监视器是使用 AUTOSTART 选项创建的，则将在第一个用户连接至数据库时激活事件监视器。而且一旦显式激活了事件监视器，每次重新激活数据库时它将自动重新启动。在显式释放事件监视器或停止实例之前，重新激活数据库时都会重新启动事件监视器。

启动表事件监视器时，事件监视器将更新 SYSCAT.EVENTMONITORS 目录表的 evmon_activates 列。将记录此更改，所以 DATABASE CONFIGURATION 将显示:

```
数据库是一致的                                    = 否
```

如果事件监视器是使用 AUTOSTART 选项创建的，并且第一个用户连接至数据库后立即断开连接从而使得数据库被释放，则会生成日志文件。

5. 要查看事件监视器处于活动状态还是不活动状态，在针对表 SYSCAT.EVENTMONITORS 的查询中发出 SQL 函数 EVENT_MON_STATE:

```
SELECT evmonname, EVENT_MON_STATE(evmonname) FROM
       syscat.eventmonitors;
```

将列示所有现有事件监视器的列表及事件监视器状态。返回值 0 指示指定的事件监视器处于不活动状态，而返回值 1 指示事件监视器处于活动状态。

6. 读取事件监视器输出。对于写至表事件监视器，这涉及检查目标表。要从 CLP 访问文件或管道事件监视器数据，请参阅“相关任务: 从命令行格式化文件或管道事件监视器输出”。
7. 为释放或关闭事件监视器，请使用 SET EVENT MONITOR 语句:

```
SET EVENT MONITOR evmonname STATE 0
```

释放事件监视器不会导致删除它。它仍然作为休眠数据库对象存在。释放事件监视器将清空其所有内容。因此，如果重新激活已释放的事件监视器，它将仅包含自重新激活后收集的信息。

在释放管道事件监视器后，关闭相应的命名管道。在 UNIX 中使用 `close()` 函数，在 Windows 2000 中则使用 `DisconnectNamedPipe()` 函数。

8. 要消除事件监视器对象，请使用 DROP EVENT MONITOR 语句:

```
DROP EVENT MONITOR evmonname
```

只能在事件监视器处于不活动状态时删除事件监视器。

在删除管道事件监视器后，删除相应的命名管道。在 UNIX 中使用 `unlink()` 函数，在 Windows 2000 中则使用 `CloseHandle()` 函数。

删除写至表事件监视器时，不会删除相关联的目标表。同样，在删除文件事件监视器时，不会删除相关联的文件。

相关概念:

- 第 55 页的『事件监视器』
- 第 85 页的『事件记录及其相应的应用程序』

相关任务:

- 第 59 页的『创建事件监视器』
- 第 72 页的『为分区数据库创建事件监视器』

- 第 74 页的『从命令行格式化文件或管道事件监视器输出』

相关参考:

- 第 75 页的『事件监视器样本输出』

创建事件监视器

事件监视器生命周期中的第一步就是创建事件监视器。在创建事件监视器之前，必须确定要将事件记录发送至 SQL 表或文件或者通过命名管道发送。对于每个事件记录目标，都将在 CREATE EVENT MONITOR SQL 语句中指定一些特定选项。CREATE EVENT MONITOR 语句的目标表必须是非分区表。在分区数据库中监视事件时也应特别注意。

先决条件:

您将需要 DBADM 权限来创建事件监视器。

过程:

- 创建表事件监视器创建表事件监视器。
- 创建文件事件监视器创建文件事件监视器。
- 创建管道事件监视器创建管道事件监视器。
- 为分区数据库创建事件监视器。

一旦创建并激活事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

相关概念:

- 第 69 页的『事件监视器文件管理』
- 第 71 页的『事件监视器命名管道管理』
- 第 62 页的『事件监视器表管理』
- 第 55 页的『事件监视器』

相关任务:

- 第 60 页的『创建表事件监视器』
- 第 66 页的『创建文件事件监视器』
- 第 70 页的『创建管道事件监视器』
- 第 72 页的『为分区数据库创建事件监视器』

相关参考:

- 『CREATE EVENT MONITOR statement』（*SQL Reference, Volume 2*）
- 第 75 页的『事件监视器样本输出』
- 第 56 页的『事件类型』

创建表事件监视器

在创建事件监视器时，必须确定所收集信息的存储位置。表事件监视器将事件记录传输至 SQL 表，它提供了文件和管道事件监视器的简单替代品，允许您轻松捕获、分析和管理事件监视数据。对于事件监视器收集的每个事件类型，将对每个关联逻辑数据组创建目标表。

表事件监视器的各种选项将在 CREATE EVENT MONITOR 语句中设置。要进一步获取对写至表事件监视器生成 CREATE EVENT MONITOR SQL 语句的帮助，可使用 db2evtbl 命令。只需要提供事件监视器的名称和期望的事件类型，就会生成 CREATE EVENT MONITOR 语句，完成所有目标表列表。然后可从 CLP 复制生成的语句，进行修改然后执行该语句。

限制:

CREATE EVENT MONITOR 语句的目标表必须是非分区表。

先决条件:

您将需要 DBADM 权限来创建表事件监视器。

过程:

1. 指示会将事件监视器数据收集在一个表（或一组文件）中。

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                         WRITE TO TABLE
   ```

   dlmon 是事件监视器的名称。
2. 指定要监视的事件的类型。可使用单个事件监视器来监视一个或多个事件类型。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO TABLE
   ```

   此事件监视器将监视 CONNECTIONS 和 DEADLOCKS WITH DETAILS 事件类型。假定上述语句由用户“riihi”发出，则目标表的派生名称和表空间如下所示:

   - riihi.connheader_dlmon
   - riihi.conn_dlmon
   - riihi.deadlock_dlmon
   - riihi.dlconn_dlmon
   - riihi.dllock_dlmon
   - riihi.control_dlmon
3. 通过调整 BUFFERSIZE 值来指定表事件监视器缓冲区的大小（以 4K 页计）:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO TABLE BUFFERSIZE 8
   ```

   8 是两个事件表缓冲区的组合容量（以 4K 页计）。求得的和为 32K 缓冲区，每个缓冲区 16K。

   BUFFERSIZE 的缺省值（和最小值）为 4 页。由于性能原因，高可用事件监视器的缓冲区应该比不活动事件监视器的缓冲区大。

4. 指示是需要分块事件监视器还是非分块事件监视器。对于分块事件监视器，如果事件缓冲区已满，则生成事件的每个代理程序将等待事件缓冲区写至表。这可能会导致数据库性能降低，原因是暂挂的代理程序和所有从属代理程序在缓冲区清空之前不能运行。使用 BLOCKED 子句来确保事件数据不会丢失:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO TABLE BUFFERSIZE 8 BLOCKED
   ```

   缺省情况下，事件监视器处于受阻状态。

   如果数据库性能比收集每个事件记录更重要，则使用非分块事件监视器。在此情况下，如果事件缓冲区已满，则生成事件的每个代理程序将不会等待事件缓冲区写至表。因此，非分块事件监视器可能会导致活动频繁的系统上的数据丢失。使用 NONBLOCKED 子句来使事件监视的性能开销降至最低:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
   ```

5. 指示需要从中收集事件记录的逻辑数据组。事件监视器将每个逻辑数据组中的数据存储在相应的表中。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO TABLE CONN, DLCONN, DLLOCK
                       BUFFERSIZE 8 NONBLOCKED
   ```

   将选择 CONN、DLCONN 和 DLLOCK 逻辑数据组。未提到的其他可用逻辑数据组包括 CONNHEADER、DEADLOCK 或 CONTROL。对于 `dlmon` 事件监视器，将不会存储与 CONNHEADER、DEADLOCK 或 CONTROL 相关的数据。

6. 指示需要从中收集数据的监视元素。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                WRITE TO TABLE CONN,
                DLCONN (EXCLUDES(agent_id, lock_wait_start_time)),
                DLLOCK (INCLUDES(lock_mode, table_name))
                BUFFERSIZE 8 NONBLOCKED
   ```

   将捕获 CONN 的所有监视元素（这是缺省行为）。对于 DLCONN，将捕获除了 agent_id 和 lock_wait_start_time 之外的所有监视元素。最后，对于 DLLOCK，lock_mode 和 table_name 是唯一捕获的监视元素。

7. 提供要创建的表的名称并指定表空间:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
   WRITE TO TABLE CONN,
   DLCONN (TABLE mydept.dlconnections,
   EXCLUDES(agent_id, lock_wait_start_time)),
   DLLOCK (TABLE dllocks, IN mytablespace,
   INCLUDES(lock_mode, table_name))
   BUFFERSIZE 8 NONBLOCKED
   ```

   假定上述语句由用户 `riihi` 发出，则目标表的派生名称和表空间如下所示:

   - CONN: riihi.conn_dlmon（在缺省表空间上）
   - DLCONN: mydept.dlconnections（在缺省表空间上）
   - DLLOCK: riihi.dllocks（在 MYTABLESPACE 表空间上）

   如果事件监视器定义者具有 USE 特权，则从 IBMDEFAULTGROUP 指定缺省表空间。如果定义者没有针对此表空间的 USE 特权，则将指定定义者对其具有 USE 特权的表空间。

8. 指示事件监视器自动释放表空间之前该表空间可以到达的充满程度:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                   WRITE TO TABLE DLCONN PCTDEACTIVATE 90
                   BUFFERSIZE 8 NONBLOCKED
```

当表空间达到容量的 90% 时，dlmon 事件监视器将自动关闭。PCTDEACTIVATE 子句只能用于 DMS 表空间。

9. 指定每次数据库启动时是否自动激活事件监视器。在缺省情况下，数据库启动时不会自动激活事件监视器。
   - 要创建在数据库启动时自动启动的事件监视器，请发出以下语句:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                        AUTOSTART NONBLOCKED
     ```
   - 要创建在数据库启动时不自动启动的事件监视器，请发出以下语句:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                        MANUALSTART
     ```

   要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE 语句。

一旦创建并激活表事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

相关概念:

- 第 62 页的『事件监视器表管理』
- 第 55 页的『事件监视器』
- 第 85 页的『事件记录及其相应的应用程序』

相关任务:

- 第 57 页的『收集关于数据库系统事件的信息』

相关参考:

- 第 56 页的『事件类型』
- 『CREATE EVENT MONITOR statement』（*SQL Reference, Volume 2*）
- 『db2evtbl - Generate event monitor target table definitions command』（*Command Reference*）

事件监视器表管理

可定义事件监视器以将其事件记录存储在 SQL 表中。为此，将 CREATE EVENT MONITOR 语句与 WRITE TO TABLE 子句配合使用。

在创建写至表事件监视器时，数据库将创建*目标表*以存储返回数据的每个逻辑数据组的记录。在缺省情况下，数据库将在事件监视器创建者模式下创建表，并且根据相应的逻辑数据组和事件监视器名称命名这些表。在每个表中，列名必须与它们表示的监视元素名称相匹配。

例如，用户 riihi 将创建捕获 STATEMENTS 事件的事件监视器:

```
CREATE EVENT MONITOR foo FOR STATEMENTS WRITE TO TABLE
```

使用 STATEMENTS 事件类型的事件监视器从 event_connheader、event_stmt 和 event_subsection 逻辑数据组收集数据。数据库创建了下列表:

- riihi.connheader_foo
- riihi.stmt_foo
- riihi.subsection_foo
- riihi.control_foo

除了提供特定于各个事件类型的逻辑数据组的表之外，还将为每个写至表事件监视器创建控制表。它在上文用 riihi.control_foo 表示。具体地说，控制表包含来自 event_start、event_db_header（仅适用于 conn_time 监视元素）和 event_overflow 逻辑数据组的事件监视器元数据。

目标表中的每个列名与一个事件监视元素标识相匹配。将忽略没有相应目标表列的事件监视元素。

写至表事件监视器目标表必须手工修剪。在高可用系统上，因为事件监视器记录的数据的高容量，事件监视器会迅速填满机器空间。与写至文件或命名管道的事件监视器不同，可将写至表事件监视器定义为仅记录特定逻辑数据组或监视元素。此功能允许您仅收集需要的相关数据，从而降低事件监视器生成的数据量。例如，以下语句定义用于捕获 TRANSACTIONS 事件的事件监视器，但它仅从 event_xact 逻辑数据组进行捕获，并且仅包括 lock_escal 监视元素:

```
CREATE EVENT MONITOR foo_lite FOR TRANSACTIONS WRITE TO TABLE
                     XACT(INCLUDES(lock_escal))
```

存在这样的情况：让事件监视器的目标表使用缺省表名称驻留在缺省表空间的缺省模式中不太理想。例如，如果预期会有大量监视数据时，您可能想要目标表在它们自己的表空间中。

可在 CREATE EVENT MONITOR 语句中指定模式、表和表空间名。模式名与表名一起提供，形成表的派生名称。

一个目标表只能被一个事件监视器使用。如果发现已经对另一个事件监视器定义了目标表，或者如果因为任何其他原因而不能创建目标表，则 CREATE EVENT MONITOR 语句会失败。

可使用可选 IN 子句在表名后添加表空间名。与 DB2 自动创建的目标表不同，如果事件监视器定义中包含某个表空间，则该表空间必须已存在。如果没有指定表空间，则将指定定义者对其具有 USE 特权的表空间。

在分区数据库环境中，写至表事件监视器将仅在包含该事件监视器表的表空间所在的数据库分区上处于活动状态。当活动事件监视器的目标表空间在特定数据库分区上不存在时，该事件监视器将在此数据库分区上取消激活，并将一个错误写入 db2diag.log 文件。

为了在检索事件监视器数据时提高性能，可为事件表创建索引。您也可以添加其他表属性，如触发器、关系完整性和约束。事件监视器将忽略它们。

例如，以下语句使用 event_connheader、event_stmt 和 event_subsection 逻辑数据组以定义用于捕获 STATEMENTS 事件的事件监视器。三个目标表中的每一个都有不同的模式、表和表空间组合:

```
CREATE EVENT MONITOR foo FOR STATEMENTS
WRITE TO TABLE CONNHEADER,
STMT (TABLE mydept.statements),
SUBSECTION (TABLE subsections, IN mytablespace)
```

假定上述语句由用户“riihi”发出，则目标表的派生名称和表空间如下所示:

- CONNHEADER: riihi.connheader_foo（在缺省表空间上）
- STMT: mydept.statements（在缺省表空间上）
- SUBSECTION: riihi.subsections（在 MYTABLESPACE 表空间上）

如果事件监视器激活时目标表不存在，则激活将继续进行，并且会忽略本来会插入至目标表的数据。相应的，如果监视元素在目标表中没有专用的列，就会被忽略。

对于活动的写至表事件监视器，可能会有存储事件记录的表空间用完容量的风险。为避免 DMS 表空间出现此风险，可定义表空间容量达到某个百分比时事件监视器将会释放。可在 CREATE EVENT MONITOR 语句的 PCTDEACTIVATE 子句中声明它。

在非分区数据库环境中，所有写至表事件监视器都会在最后一个应用程序终止（并且数据库未显式激活）时释放。在分区数据库环境中，写至表事件监视器将在目录分区释放时释放。

下表提供缺省目标表名，这些表名是按对其返回表名的事件类型排序的。

表 10. 写至表事件监视器目标表

事件类型	目标表名	可用信息
DEADLOCKS	CONNHEADER DEADLOCK DLCONN CONTROL	连接元数据 死锁数据 死锁涉及的应用程序和锁定 事件监视器元数据
DEADLOCKS WITH DETAILS	CONNHEADER DEADLOCK DLCONN DLLOCK CONTROL	连接元数据 死锁数据 死锁涉及的应用程序 死锁涉及的锁定 事件监视器元数据
DEADLOCKS WITH DETAILS HISTORY	CONNHEADER DEADLOCK DLCONN DLLOCK STMTHIST CONTROL	连接元数据 死锁数据 死锁涉及的应用程序 死锁涉及的锁定 工作单元中的先前语句列表 事件监视器元数据
DEADLOCKS WITH DETAILS HISTORY DATA VALUES	CONNHEADER DEADLOCK DLCONN DLLOCK STMTHIST STMTVALS CONTROL	连接元数据 死锁数据 死锁涉及的应用程序 死锁涉及的锁定 工作单元中的先前语句列表 STMTHIST 表中的语句的输入数据值 事件监视器元数据
STATEMENTS	CONNHEADER STMT SUBSECTION CONTROL	连接元数据 语句数据 特定于子节的语句数据 事件监视器元数据
TRANSACTIONS	CONNHEADER XACT CONTROL	连接元数据 事务数据 事件监视器元数据
CONNECTIONS	CONNHEADER CONN CONTROL CONMEMUSE	连接元数据 连接数据 事件监视器元数据 内存池元数据
DATABASE	DBCONTROL DBMEMUSE	数据库管理器数据 事件监视器元数据 内存池元数据
BUFFERPOOLS	BUFFERPOOLCONTROL	缓冲池数据 事件监视器元数据
TABLESPACES	TABLESPACECONTROL	表空间数据 事件监视器元数据
TABLES	TABLECONTROL	表数据 事件监视器元数据

未对写至表事件监视器收集下列逻辑数据组:

- event_log_stream_header
- event_log_header
- event_dbheader（仅收集 conn_time 监视元素）

事件监视器表中的每一列的数据类型对应于该列表示的监视元素的数据类型。以下是一组数据类型映射，这些映射使监视元素（在 sqlmon.h 中）的原始系统监视器数据类型与表列的 SQL 数据类型相对应。

表 11. 系统监视器数据类型映射

系统监视器数据类型	SQL 数据类型
SQLM_TYPE_STRING	CHAR[n]、VARCHAR[n] 或 CLOB[n]
SQLM_TYPE_U8BIT 和 SQLM_TYPE_8BIT	SMALLINT、INTEGER 或 BIGINT
SQLM_TYPE_U16BIT 和 SQLM_TYPE_16BIT	SMALLINT、INTEGER 或 BIGINT
SQLM_TYPE_U32BIT 和 SQLM_TYPE_32BIT	INTEGER 或 BIGINT
SQLM_TYPE_U64BIT 和 SQLM_TYPE_64BIT	BIGINT
SQLM_TIMESTAMP	TIMESTAMP
SQLM_TIME	BIGINT
SQLCA: SQLERRMC	VARCHAR[72]
SQLCA: SQLSTATE	CHAR[5]
SQLCA: SQLWARN	CHAR[11]
SQLCA: 其他字段	INTEGER 或 BIGINT
SQLM_TYPE_HANDLE	BLOB[n]

注:

1. 并非所有列都是空值。
2. 因为带有 CLOB 列的表的性能低于带有 VARCHAR 列的表，所以在指定 stmt evmGroup（或在使用带有详细信息的死锁时指定 dlconn evmGroup）时考虑使用 TRUNC 关键字。
3. SQLM_TYPE_HANDLE 用于表示编译环境句柄对象。

相关概念:

- 第 55 页的『事件监视器』
- 第 124 页的『事件类型至逻辑数据组的映射』
- 第 70 页的『写入表操作和文件事件监视器缓冲』

相关任务:

- 第 60 页的『创建表事件监视器』

相关参考:

- 第 401 页的『event_monitor_name － 事件监视器名称』

创建文件事件监视器

在创建事件监视器时，必须确定所收集信息的存储位置。文件事件监视器将事件记录流输送至一系列 8 字符编号的文件，这些文件的扩展名为“EVT”（如 00000000.evt、00000001.evt 和 00000002.evt）。即使数据被分为较小的数据块，也应考虑将数据作为一个逻辑文件（即，数据流的开头是文件 00000000.evt 中的第一个字节；数据流的结尾是文件 nnnnnnnn.evt 的最后一个字节）。事件监视器永远不会让单个事件记录跨越两个文件。

文件事件监视器及其选项将由 `CREATE EVENT MONITOR` 语句定义。

先决条件:

您将需要 DBADM 权限来创建文件事件监视器。

过程:

1. 指定会将事件监视器数据收集在一个文件或一组文件中，并提供存储事件文件的目录位置。

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                        WRITE TO FILE '/tmp/dlevents'
   ```

   dlmon 是事件监视器的名称。

   /tmp/dlevents 是目录路径（在 UNIX 上）的名称，事件监视器会将事件文件写至该目录。
2. 指定要监视的事件的类型。可使用单个事件监视器来监视一个或多个事件类型。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO FILE '/tmp/dlevents'
   ```

   此事件监视器将监视 CONNECTIONS 和 DEADLOCKS WITH DETAILS 事件类型。
3. 通过调整 BUFFERSIZE 值来指定文件事件监视器缓冲区的大小（以 4K 页计）:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
   ```

   8 是两个事件文件缓冲区的容量（以 4K 页计）。

   BUFFERSIZE 的缺省值（和最小值）为 4。由于性能原因，高可用事件监视器的缓冲区应该比不活动事件监视器的缓冲区大。
4. 指示是需要分块事件监视器还是非分块事件监视器。对于分块事件监视器，如果事件缓冲区已满，则生成事件的每个代理程序将等待事件缓冲区写至文件。这可能会导致数据库性能降低，原因是暂挂的代理程序和所有从属代理程序在缓冲区清空之前不能运行。使用 BLOCKED 子句来确保事件数据不会丢失:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                        BLOCKED
   ```

   缺省情况下，事件监视器处于受阻状态。

   如果数据库性能比收集每个事件记录更重要，则使用非分块事件监视器。在此情况下，如果事件缓冲区已满，则生成事件的每个代理程序将不会等待事件缓冲区写至文件。因此，非分块事件监视器可能会导致活动频繁的系统上的数据丢失。使用 NONBLOCKED 子句来使事件监视的性能开销降至最低:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                        NONBLOCKED
   ```
5. 指定可对每个事件监视器收集的事件文件的最大数目。如果达到此限制，则事件监视器将释放自身。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                        NONBLOCKED MAXFILES 5
   ```

   5 是将要创建的事件文件的最大数目。

还可指定不限制事件监视器可创建的事件文件数:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                     WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                     NONBLOCKED MAXFILES NONE
```

6. 指定事件监视器创建的每个事件文件的最大大小（以 4K 页计）。如果达到此限制，则创建新文件。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                     WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                     NONBLOCKED MAXFILES 5 MAXFILESIZE 32
```

32 是事件文件可包含的最大 4K 页数。

此值必须大于 BUFFERSIZE 参数指定的值。还可指定不限制事件文件的大小:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                     WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                     NONBLOCKED MAXFILES NONE MAXFILESIZE NONE
```

7. 指定每次数据库启动时是否自动激活事件监视器。在缺省情况下，数据库启动时不会自动激活事件监视器。
   - 要创建在数据库启动时自动启动的事件监视器，请发出以下语句:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                     WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                     NONBLOCKED AUTOSTART
```

   - 要创建在数据库启动时不自动启动的事件监视器，请发出以下语句:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                     WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                     NONBLOCKED MANUALSTART
```

要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE 语句。

一旦创建并激活文件事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

相关概念:

- 第 69 页的『事件监视器文件管理』
- 第 55 页的『事件监视器』
- 第 124 页的『事件类型至逻辑数据组的映射』
- 第 70 页的『写入表操作和文件事件监视器缓冲』

相关任务:

- 第 57 页的『收集关于数据库系统事件的信息』
- 第 72 页的『为分区数据库创建事件监视器』
- 第 74 页的『从命令行格式化文件或管道事件监视器输出』

相关参考:

- 第 75 页的『事件监视器样本输出』
- 第 56 页的『事件类型』

事件监视器文件管理

文件事件监视器允许事件监视器将其事件记录存储在文件中。事件监视器的所有输出将存储在 CREATE EVENT MONITOR 语句的 FILE 参数提供的目录中。如果此目录不存在，则 DB2 将不会创建它。在激活监控器之前，该目录必须存在，否则 SET EVENT MONITOR 命令将返回错误。当第一次激活文件事件监视器时，将在此目录中创建控制文件 db2event.ctl。不要除去或修改此文件。

在缺省情况下，事件监视器会将跟踪写至称为 00000000.evt 的单个文件。只要文件系统上还有空间，此文件就会一直增长。如果使用 CREATE EVENT MONITOR 语句的 MAXFILESIZE 参数指定了文件大小限制，则在文件已满时，输出将自动导向下一个文件。因此，活动文件就是具有最高编号的文件。

还可通过使用 CREATE EVENT MONITOR 语句的 MAXFILES 参数来限制整个事件监视器跟踪的最大大小。当文件数达到 MAXFILES 定义的最大数目时，事件监视器将释放自身，并且会向管理通知日志写入以下消息。

```
DIA1601I 事件监视器 monitor-name 在达到预设的 MAXFILES 和 MAXFILESIZE 限制时释放。
```

可通过除去已满的文件来避免出现此情况。当事件监视器仍在运行时，可以除去活动文件之外的任何事件文件。

如果文件事件监视器用完了磁盘空间，则在管理通知日志中记录系统错误级别消息后，它会关闭自身。

重新启动文件事件监视器时，它会擦除任何现有数据或追加的新数据。在 CREATE EVENT MONITOR 语句中指定此选项，并可创建 APPEND 监视器或 REPLACE 监视器。APPEND 是缺省选项。APPEND 事件监视器开始在上次使用的文件结尾写入。如果已除去该文件，则会使用下一个顺序文件编号。重新启动追加事件监视器时，仅生成 start_event。仅对第一次激活生成事件日志头和数据库头。替换（REPLACE）事件监视器总是会删除现有事件文件并开始在 00000000.evt 中写入。

您可能想要在事件监视器活动时处理监视器数据。这是可行的，而且在处理完文件后可以删除它，从而释放空间以供后续监视数据使用。除非停止并重新启动事件监视器，否则不能强制事件监视器切换至下一个文件。它必须也处于 APPEND 方式。为记录在活动文件中处理的事件，可创建一个应用程序，它只记录已处理的上一个记录的文件编号和位置。下一次处理跟踪时，该应用程序只需搜索该文件的位置。

相关概念:

- 第 85 页的『事件监视器自描述数据流』
- 第 55 页的『事件监视器』
- 第 124 页的『事件类型至逻辑数据组的映射』
- 第 7 页的『系统监视器输出：自描述数据流』
- 第 70 页的『写入表操作和文件事件监视器缓冲』

相关任务:

- 第 57 页的『收集关于数据库系统事件的信息』
- 第 66 页的『创建文件事件监视器』

相关参考:

• 第 75 页的『事件监视器样本输出』

写入表操作和文件事件监视器缓冲

事件监视器进程使用两个内部缓冲区来对它的记录进行缓存，然后再将这些记录写入文件或表。缓冲区满时，将自动写记录。因此，通过指定较大的缓冲区以减少磁盘访问次数，可以改善高吞吐量事件监视器的监视性能。为了强制事件监视器清空其缓冲区，必须释放该事件监视器，或者使用 FLUSH EVENT MONITOR 语句来清空缓冲区。

对于受阻事件监视器来说，当它的两个缓冲区都变满时，它将暂挂正在发送监视器数据的数据库进程。这是为了确保在受阻事件监视器活动期间不会删除任何事件记录。在将缓冲区内容写入文件或表之前，暂挂的数据库进程以及任何从属数据库进程都无法运行。根据工作负载类型以及 I/O 设备速度的不同，这会对性能产生严重影响。缺省情况下，事件监视器处于受阻状态。

对于非受阻事件监视器来说，如果从代理程序接收监视器数据的速度高于事件监视器写数据的速度时，它将删除那些数据。这样就可以避免事件监视操作影响其他数据库活动的性能。

删除了事件记录的事件监视器会生成溢出事件。此事件指定了监视器删除事件的开始时间和停止时间以及在该时间段内删除的事件数。事件监视器有可能对要报告的暂挂溢出终止或释放。如果发生这种情况，就会将以下消息写入管理日志:

```
DIA2503I 事件监视器 monitor-name 在释放时有暂挂的溢出记录。
```

各个事件记录也会发生丢失事件监视数据的情况。如果事件记录长度超过事件缓冲区大小，在缓冲区中装不下的数据就会被截断。例如，如果正在捕获 stmt_text 监视元素并且连接到被监视数据库的应用程序发出了很长的 SQL 语句，就会发生数据截断的情况。如果需要捕获所有事件记录信息，请指定较大的缓冲区。记住，较大的缓冲区会降低写入文件或表的频率。

相关概念:

• 第 69 页的『事件监视器文件管理』
• 第 62 页的『事件监视器表管理』
• 第 55 页的『事件监视器』

相关任务:

• 第 66 页的『创建文件事件监视器』
• 第 60 页的『创建表事件监视器』

创建管道事件监视器

在创建事件监视器时，必须确定所收集信息的存储位置。管道事件监视器直接将事件记录从事件监视器传输至命名管道。在事件监视器写入事件数据时，将由监视应用程序迅速读取管道中的数据。如果事件监视器无法将数据写至管道（例如，因为管道已满），则监视器数据将会丢失。

管道事件监视器将使用 `CREATE EVENT MONITOR` 语句定义。

先决条件:

您将需要 DBADM 权限来创建管道事件监视器。

过程:

1. 指示会将事件监视器数据引导至命名管道。

```
CREATE EVENT MONITOR dlmon FOR eventtype
                 WRITE TO PIPE '/home/riihi/dlevents'
```

   dlmon 是事件监视器的名称。

   /home/riihi/dlevents 是命名管道（在 UNIX 上）的名称，事件监视器会将事件记录引导至该命名管道。CREATE EVENT MONITOR 语句支持 UNIX 和 Windows 管道命名语法。

   激活事件监视器时，CREATE EVENT MONITOR 语句中指定的命名管道必须存在并且打开。如果指定事件监视器将自动启动，则在创建事件监视器之前命名管道必须存在。

2. 指定要监视的事件的类型。可使用单个事件监视器来监视一个或多个事件类型。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                 WRITE TO PIPE '/home/riihi/dlevents'
```

   此事件监视器将监视 CONNECTIONS 和 DEADLOCKS WITH DETAILS 事件类型。

3. 指定每次数据库启动时是否自动激活事件监视器。在缺省情况下，数据库启动时不会自动激活事件监视器。
   - 要创建在数据库启动时自动启动的事件监视器，请发出以下语句:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                 WRITE TO PIPE '/home/riihi/dlevents'
                 AUTOSTART
```

   - 要创建在数据库启动时不自动启动的事件监视器，请发出以下语句:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                 WRITE TO PIPE '/home/riihi/dlevents'
                 MANUALSTART
```

   要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE 语句。

一旦创建并激活管道事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

相关概念:

- 第 71 页的『事件监视器命名管道管理』
- 第 85 页的『事件监视器自描述数据流』
- 第 7 页的『系统监视器输出：自描述数据流』

相关参考:

- 第 75 页的『事件监视器样本输出』

事件监视器命名管道管理

管道事件监视器允许通过命名管道处理事件监视器数据流。如果需要实时处理事件记录，则最好使用管道事件监视器。另一个很重要的优点是，应用程序在读完管道时可以忽略不想要的数据，从而可以大幅降低存储器需求。

在 AIX 上，可使用 mkfifo 命令来创建命名管道。在 Linux 和其他 UNIX 类型（如 Solaris 操作系统）上，使用 pipe() 例程。在 Windows 上，可通过使用 CreateNamedPipe() 例程来创建命名管道。

将数据引导至管道时，I/O 总是会分块并且唯一的缓冲将由管道执行。在事件监视器写入事件数据时，将由监视应用程序迅速读取管道中的数据。如果事件监视器无法将数据写至管道（例如，因为管道已满），则监视器数据将会丢失。

此外，命名管道中必须有足够的空间用来处理入局事件记录。如果应用程序从命名管道读取数据时不够快，管道将填满并溢出。管道缓冲区越小，溢出的机率越大。

当发生管道溢出时，监视器将创建溢出事件记录以指示发生溢出。事件监视器不会关闭，但监视器数据会丢失。如果释放监视器时存在明显的溢出事件记录，则会记录诊断消息。否则，溢出事件记录可能时将写至管道。

如果操作系统允许定义管道缓冲区的大小，则使用的管道缓冲区至少应该为 32K。对于高容量事件监视器，应将监视应用程序的进程优先级设置为等于或高于代理进程优先级。

相关概念:

- 第 85 页的『事件监视器自描述数据流』
- 第 55 页的『事件监视器』
- 第 124 页的『事件类型至逻辑数据组的映射』
- 第 7 页的『系统监视器输出：自描述数据流』

相关任务:

- 第 57 页的『收集关于数据库系统事件的信息』
- 第 59 页的『创建事件监视器』

相关参考:

- 第 75 页的『事件监视器样本输出』

为分区数据库创建事件监视器

在分区数据库系统上创建文件或管道事件监视器时，需要确定想要收集的监视数据的作用域。事件监视器使用操作系统进程或线程来写入事件记录。运行此进程或线程的数据库分区称为监视器分区。如果文件和管道事件监视器在监视器分区上以局部方式运行，或者在运行 DB2 数据库管理器的任何分区上以全局方式运行，则这些事件监视器可充当监视事件。全局事件监视器将在监视分区上写入单个跟踪，它包含所有分区的活动。不管事件监视器是局部的还是全局的，都被称为监视作用域。

监视器分区和监视器作用域都将使用 CREATE EVENT MONITOR 语句指定。

仅当监视器分区处于活动状态时，才能激活事件监视器。如果 SET EVENT MONITOR 语句用于激活事件监视器但监视器分区尚未处于活动状态，则将在下一次启动监视器分区时激活事件监视器。而且，在显式释放事件监视器或显式释放实例之后，将自动激活事件监视器。例如，在数据库分区 0 上:

```
db2 connect to sample
db2 create event monitor foo ... on dbpartitionnum 2
db2 set event monitor foo state 1
```

在运行上述命令后，每次数据库 sample 在数据库分区 2 上激活时都将自动激活事件监视器 foo。发出 db2 set event monitor foo state 0 或停止分区 2 之前，会一直进行此自动激活。

局部或全局作用域的注释不适用于写至表事件监视器。激活写至表事件监视器时，事件监视器在所有分区上运行。（更具体地说，事件监视器进程将在属于数据库分区组的分区上运行，而目标表位于这些数据库分区组上。）运行事件监视器进程的每个分区具有同一组目标表。因为是从各个分区的角度表示监视数据，所以这些表中的数据将会有所不同。可通过发出 SQL 语句来访问每个分区的事件监视器目标表中的期望值，以获取来自所有分区的聚集值。

每个目标表的第一列称为 PARTITION_KEY，并且被用作表的分区键。此列的值将被选中，以便每个事件监视器进程将数据插入到运行该进程的数据库分区中；即，插入操作将在运行事件监视器进程的数据库分区本地执行。在任何数据库分区上，PARTITION_KEY 字段都将包含相同的值。这意味着，如果删除数据分区并且重新分布数据，则被删除数据库分区上的所有数据都将转至另一数据库分区而不是平均分布。因此，在除去数据库分区之前，考虑删除该数据库分区上的所有表行。

此外，可对每个表定义名为 PARTITION_NUMBER 的列。此列包含插入数据的分区的编号。与 PARTITION_KEY 列不同，PARTITION_NUMBER 列不是必需的。

用来定义目标表的表空间必须在所有分区上存在，事件监视器数据将写至这些分区。如果不遵守此规则，则不会使用事件监视器将记录写至不存在表空间的登录分区。事件仍将写至存在表空间的分区，并且不返回任何错误。此行为允许用户通过创建只在特定分区上存在的表空间，以选择要监视的分区子集。

在写至表事件监视器激活期间，FIRST_CONNECT 和 EVMON_START 的 CONTROL 表行将仅插入至目录数据库分区。这要求目录数据库分区上存在控制表的表空间。如果目录数据库分区上不存在该表空间，则不会执行这些插入。如果激活写至表事件监视器时分区尚未激活，则将在激活事件监视器之前激活该分区。在此情况下，数据库激活行为就好像 SQL CONNECT 语句在所有分区上激活了该数据库一样。

注: 详细的死锁连接事件中的锁定列表将仅包含这样的锁定，这些锁定是应用程序在等待死锁的分区上挂起的。例如，如果涉及死锁的应用程序正在节点 20 上等待锁定，则只有该应用程序在节点 20 上挂起的锁定才会包括在列表中。

先决条件:

您将需要 DBADM 权限来为分区数据库创建事件监视器。

过程:

1. 指定要监视的分区。

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                       WRITE TO FILE '/tmp/dlevents'
                       ON PARTITION 3
   ```

   dlmon 表示事件监视器的名称。

   /tmp/dlevents 是目录路径（在 UNIX 上）的名称，事件监视器会将事件文件写至该目录。

   3 表示要监视的分区号。

2. 指定是在局部作用域还是全局作用域收集事件监视器数据。要从所有分区收集事件监视器报告，则发出以下语句:

```
CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                     WRITE TO FILE '/tmp/dlevents'
                     ON PARTITION 3 GLOBAL
```

   只有死锁和与带有详细信息事件监视器的死锁才能定义为 GLOBAL。所有分区会将与死锁有关的事件记录报告至分区 3。

   要仅从局部分区收集事件监视器报告，则发出以下语句:

```
CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                     WRITE TO FILE '/tmp/dlevents'
                     ON PARTITION 3 LOCAL
```

   这是分区数据库中的文件和管道事件监视器的缺省行为。对于写至表事件监视器，将忽略 LOCAL 和 GLOBAL 子句。
3. 可复查现有事件监视器的监视器分区和作用域值。为此，请使用以下语句查询 SYSCAT.EVENTMONITORS 表:

```
SELECT EVMONNAME, NODENUM, MONSCOPE FROM SYSCAT.EVENTMONITORS
```

一旦创建并激活事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

相关概念:

- 第 6 页的『计数器状态和可视性』
- 第 55 页的『事件监视器』
- 第 124 页的『事件类型至逻辑数据组的映射』

相关任务:

- 第 66 页的『创建文件事件监视器』
- 第 60 页的『创建表事件监视器』

相关参考:

- 第 75 页的『事件监视器样本输出』
- 第 56 页的『事件类型』

从命令行格式化文件或管道事件监视器输出

文件或管道事件监视器的输出是一个逻辑数据分组二进制流。可使用 db2evmon 命令从命令行格式化此数据流。此高效工具从事件监视器的文件或管道读取事件记录，然后将它们写至屏幕（标准输出）。

可通过提供事件文件的路径或提供数据库名称和事件监视器名称，以指示想要格式化的事件监视器输出。

先决条件:

除非连接至数据库，否则不需要任何权限，如果连接至数据库，则需要下列其中一个权限:

- SYSADM
- SYSCTRL

- SYSMAINT
- DBADM

过程:

要格式化事件监视器输出:

- 指定包含事件监视器文件的目录:

  ```
  db2evmon -path '/tmp/dlevents'
  ```

  `/tmp/dlevents` 表示（UNIX）路径。
- 指定数据库和事件监视器名称:

  ```
  db2evmon -db 'sample' -evm 'dlmon'
  ```

  `sample` 表示事件监视器所属的数据库。

  `dlmon` 表示事件监视器。

相关概念:

- 第 69 页的『事件监视器文件管理』
- 第 71 页的『事件监视器命名管道管理』
- 第 85 页的『事件监视器自描述数据流』

相关任务:

- 第 66 页的『创建文件事件监视器』
- 第 70 页的『创建管道事件监视器』

事件监视器样本输出

为说明事件监视的特征，以下是死锁监视方案的示例。要实施此方案，需要三个 DB2 CLP 窗口。我们将第一个 CLP 称为监视器会话，余下两个 CLP 称为应用程序 1 和应用程序 2。还需要 DB2 SAMPLE 数据库。

注: 可使用下列其中一个步骤创建并填充 SAMPLE 数据库:

- UNIX: `sqllib/bin/db2sampl`
- Windows: `sqllib\bin\db2sampl.exe`

从监视器会话中定义事件监视器，该事件监视器将记录表数据和数据库的连接之间发生的死锁数。

监视器会话

```
db2 connect to sample
db2 "create event monitor dlmon for tables, deadlocks with details
     write to file 'c:\dlmon'"
mkdir c:\dlmon
db2 "set event monitor dlmon state 1"
```

现在，使用数据库的两个应用程序将进入死锁状态。死锁是这样一种情况：每个应用程序挂起另一应用程序继续处理所必需的锁定。最终由 DB2 死锁检测器组件检测并解决死锁，这将回滚其中一个事务。因此只有一个应用程序成功完成事务。下图显示了此方案。

应用程序 1

```
db2 connect to sample
db2 +c "insert into staff values(26, 'Simpson',
        2, 'Mgr', 13, 35000, 0)"
```

应用程序 1 现在挂起针对 STAFF 表的某行的互斥锁定。

注: 上面使用的 +c 选项将给定 CLP 命令的自动落实设置为 OFF。

应用程序 2

```
db2 connect to sample
db2 +c "insert into department values('7G', 'Safety',
        '1', 'A00', NULL)"
```

应用程序 2 现在挂起针对 DEPARTMENT 表的某行的互斥锁定。

应用程序 1

```
db2 +c "select deptname from department"
```

假定存在游标稳定性，在访存 department 表的各行时应用程序 1 需要针对每行的共享锁定，但因为应用程序 2 具有针对最后一行的互斥锁定，所以应用程序 1 不能获取针对该行的锁定。应用程序 1 进入 LOCK WAIT 状态，等待该锁定被释放。

应用程序 2

```
db2 +c "select name from staff"
```

应用程序 2 也会进入 LOCK WAIT 状态，等待应用程序 1 释放针对 staff 表的最后一行的互斥锁定。

这些应用程序现在处于死锁状态。因为每个应用程序都挂起另一资源继续进行所需的资源，所以此等待永远不会结束。最后，死锁检测器检查死锁并选择一个牺牲者以进行回滚:

应用程序 2

```
SQLN0991N 因为死锁超时，所以当前事务已回滚。原因码为“2”。SQLSTATE=40001
```

此时事件监视器将死锁事件记录至目标。应用程序 1 现在可以继续进行:

应用程序 1

```
DEPTNAME
-----------------------------
PLANNING
INFORMATION CENTER
. . .
SOFTWARE SUPPORT
选择了 9 个记录
```

因为事件监视器缓冲其输出并且此方案未生成足够的事件记录来填充缓冲区，所以事件监视器值将强制写至事件监视器输出写程序。为生成表事件数据（它们是在释放数据库时生成的），应用程序 1、应用程序 2 和监视器会话将与数据库断开连接。

应用程序 1

```
db2 connect reset
```

应用程序 2

```
db2 connect reset
```

在监视器会话 CLP 中与数据库断开连接后，将以二进制文件的形式写入事件跟踪。现在可使用 db2evmon 工具来对其进行格式化:

监视器会话

```
db2 connect resetdb2evmon -path c:\dlmon
```

事件监视器使用的逻辑数据分组将按以下四个不同级别排序和显示：监视器、序言、内容和结尾。

监视器:

将对所有事件监视器生成监视器级别的信息。它包含事件监视器元数据。

只有返回 SQLM_DBMON_VERSION6、SQLM_DBMON_VERSION7 或 SQLM_DBMON_VERSION8 中的一个版本的事件监视器才使用自描述数据流。版本 6 之前的输出必须使用版本 5 方法读取。有关这些静态大小的结构的信息，请参阅 sqlmon.h 文件。

序言:

在激活事件监视器时将生成序言信息。

```
--------------------------------------------------------------------------
                            EVENT LOG HEADER
  事件监视器名称: DLMON
  服务器产品标识: SQL08020
  事件监视器数据的版本: 7
  字节顺序: LITTLE ENDIAN
  DB2 实例中的节点数: 1
  数据库的代码页: 1252
  数据库的地域代码: 1
  服务器实例名称: DB2
--------------------------------------------------------------------------

--------------------------------------------------------------------------
  数据库名称: SAMPLE
  数据库路径: D:\DB2\NODE0000\SQL00001\
  第一次连接时间戳记: 11/25/2003 13:07:32.649113
  事件监视器启动时间: 11/25/2003 13:07:52.292867
--------------------------------------------------------------------------
```

内容:

特定于事件监视器的指定事件类型的信息出现在内容部分。此部分中记录的事件包含对衍生这些事件（应用程序句柄或应用程序标识）的应用程序的引用。如果跟踪多个应用程序中的事件，则使用应用程序标识来记录各种事件。与内容部分中的所有其他事件不同，溢出事件并未对应特定事件类型。它将跟踪丢失的记录数：它们是在系统不能与（非分块）事件监视器保持一致时生成的。在此示例中存在死锁事件，这些事件将由先前语句导致的死锁状态衍生。

```
3) 连接头事件 ...
  应用程序句柄: 12
  应用程序标识: *LOCAL.DB2.0063C5180732
  应用程序序号: 0001
  DRDA AS 关联标记: *LOCAL.DB2.0063C5180732
  应用程序序号: 0001
  DRDA AS 关联标记: *LOCAL.DB2.0063C5180732
  程序名: db2bp.exe
  授权标识: RIIHI
  执行标识: RIIHI
  代码页标识: 1252
  地域代码: 1
  客户机进程标识: 312
  客户机数据库别名: SAMPLE
  客户机产品标识: SQL08020
  客户机平台: 未知
  客户机通信协议: 本地
  客户机网络名:
  连接时间戳记: 11/25/2003 13:07:32.649113

4) 连接头事件 ...
  应用程序句柄: 13
  应用程序标识: *LOCAL.DB2.00FE05180800
  应用程序序号: 0001
  DRDA AS 关联标记: *LOCAL.DB2.00FE05180800
  程序名: db2bp.exe
  授权标识: RIIHI
  执行标识: RIIHI
  代码页标识: 1252
  执行标识: RIIHI
  代码页标识: 1252
  地域代码: 0
  客户机进程标识: 2144
  客户机数据库别名: SAMPLE
  客户机产品标识: SQL08020
  客户机平台: 未知
  客户机通信协议: 本地
```

```
  客户机网络名:
  连接时间戳记: 11/25/2003 13:08:00.897979

5) 死锁事件 ...
  死锁标识: 1
  带有死锁的应用程序数目: 2
  死锁检测时间: 11/25/2003 13:09:02.831833
  已回滚应用程序参与者编号: 2
  已回滚应用程序标识: *LOCAL.DB2.00FE05180800
  已回滚应用程序序号: 0001

6) 死锁的连接 ...
  死锁标识: 1
  参与者编号: 2
  挂起该锁定的参与者编号: 1
  应用程序标识: *LOCAL.DB2.00FE05180800
  挂起该锁定的参与者编号: 1
  应用程序标识: *LOCAL.DB2.00FE05180800
  应用程序序号: 0001
  挂起该锁定的连接的应用程序标识: *LOCAL.DB2.00FE05180800
  挂起该锁定的连接的序号: 0001
  锁定等待开始时间:
      锁定名称                    : 0x02000300280000000000000052
      锁定属性                    : 0x00000000
      释放标志                    : 0x00000001
      锁定计数                    : 1
      挂起计数                    : 0
  当前方式: 无
  死锁检测时间: 11/25/2003 13:09:02.832048
  被等待的锁定的表: STAFF
  被等待的锁定的模式: RIIHI

  被等待的锁定的表空间: USERSPACE1
  锁定类型: 行
  锁定方式: X – 独占
  应用程序对锁定请求的方式: NS – 共享(及下一键共享)
  发生锁定的节点: 0
  锁定对象名: 40
  应用程序句柄: 13
  死锁的语句:
    类型: 动态
    操作: 访存
    段: 201
    创建者: NULLID
    程序包: SQLC2E03
    游标: SQLCUR201
    游标正在分块: FALSE
    文本: select name from staff
  锁定列表:
      锁定名称                    : 0x010000000100000001004C0056
      锁定属性                    : 0x00000000
      释放标志                    : 0x40000000
      锁定计数                    : 1
      挂起计数                    : 0
      锁定对象名                  : 0
      对象类型                    : 内部 – 偏差
      方式                        : S – 共享

  锁定名称                        : 0x020004000D0000000000000052
      锁定属性                    : 0x00000000
      释放标志                    : 0x40000000
      锁定计数                    : 1
      挂起计数                    : 0
      锁定计数                    : 1
      挂起计数                    : 0
      锁定对象名                  : 13
      对象类型                    : 行
```

```
      表空间名称                  : USERSPACE1
      表模式                      : RIIHI
      表名                        : DEPARTMENT
      方式                        : X - 独占

      锁定名称                    : 0x94928D848F9F949E7B89505241
      锁定属性                    : 0x00000000
      释放标志                    : 0x40000000
      锁定计数                    : 1
      挂起计数                    : 0
      锁定对象名                  : 0
      对象类型                    : 内部 - 计划
      方式                        : S - 共享

      锁定名称                    : 0x96A09A989DA09A7D8E8A6C7441
      锁定属性                    : 0x00000000
      释放标志                    : 0x40000000
      锁定计数                    : 1
      挂起计数                    : 0
      锁定对象名                  : 0
      挂起计数                    : 0
      锁定对象名                  : 0
      对象类型                    : 内部 - 计划
      方式                        : S - 共享

      锁定名称                    : 0x02000400000000000000000054
      锁定属性                    : 0x00000000
      释放标志                    : 0x40000000
      锁定计数                    : 1
      挂起计数                    : 0
      锁定对象名                  : 4
      对象类型                    : 表
      表空间名称                  : USERSPACE1
      表模式                      : RIIHI
      表名                        : DEPARTMENT
      方式                        : IX - 意向互斥

      锁定名称                    : 0x02000300000000000000000054
      锁定属性                    : 0x00000000
      释放标志                    : 0x00000001
      锁定计数                    : 1
      挂起计数                    : 0
      锁定对象名                  : 3
      对象类型                    : 表
      锁定对象名                  : 3
      对象类型                    : 表
      表空间名称                  : USERSPACE1
      表模式                      : RIIHI
      表名                        : STAFF
      方式                        : IS - 意向共享

  挂起的锁定数: 6
  列表中的锁定数: 6

7) 死锁的连接 ...
  死锁标识: 1
  参与者编号: 1
  挂起该锁定的参与者编号: 2
  应用程序标识: *LOCAL.DB2.0063C5180732
  应用程序序号: 0002
  挂起该锁定的连接的应用程序标识: *LOCAL.DB2.0063C5180732
  挂起该锁定的连接的序号: 0002
  锁定等待开始时间:
  锁定名称                        : 0x020004000D0000000000000052
      锁定属性                    : 0x00000000
      释放标志                    : 0x00000001
      锁定计数                    : 1
```

```
    挂起计数                    : 0
    锁定计数                    : 1
    挂起计数                    : 0
当前方式: 无
死锁检测时间: 11/25/2003 13:09:02.968324
被等待的锁定的表: DEPARTMENT
被等待的锁定的模式: RIIHI
被等待的锁定的表空间: USERSPACE1
锁定类型: 行
锁定方式: X - 独占
应用程序对锁定请求的方式: NS - 共享(及下一键共享)
发生锁定的节点: 0
锁定对象名: 13
应用程序句柄: 12
死锁的语句:
  类型: 动态
  操作: 访存
  段: 201
  创建者: NULLID
  程序包: SQLC2E03
  游标: SQLCUR201
  游标正在分块: FALSE
  文本: select deptname from department
锁定列表:
锁定名称                        : 0x020004000D0000000000000052
    锁定属性                    : 0x00000000
    释放标志                    : 0x00000001
    锁定计数                    : 1
    挂起计数                    : 0
    锁定对象名                  : 13
    对象类型                    : 行
    表空间名称                  : USERSPACE1
    表模式                      : RIIHI
    表名                        : DEPARTMENT
    方式                        : NS - 共享(及下一键共享)

    锁定名称                    : 0x01000000010000000100640056
    锁定属性                    : 0x00000000
    释放标志                    : 0x40000000
    锁定计数                    : 1
    挂起计数                    : 0
    锁定对象名                  : 0
    对象类型                    : 内部 - 偏差
    方式                        : S - 共享

    锁定名称                    : 0x02000300280000000000000052
    锁定属性                    : 0x00000000
    锁定名称                    : 0x02000300280000000000000052
    锁定属性                    : 0x00000000
    释放标志                    : 0x40000000
    锁定计数                    : 1
    挂起计数                    : 0
    锁定对象名                  : 40
    对象类型                    : 行
    表空间名称                  : USERSPACE1
    表模式                      : RIIHI
    表名                        : STAFF
    方式                        : X - 独占

    锁定名称                    : 0x94928D848F9F949E7B89505241
    锁定属性                    : 0x00000000
    释放标志                    : 0x40000000
    锁定计数                    : 1
    挂起计数                    : 0
    锁定对象名                  : 0
    对象类型                    : 内部 - 计划
    方式                        : S - 共享
```

```
      锁定名称                          : 0x02000400000000000000000054
      锁定属性                          : 0x00000000
      释放标志                          : 0x00000001
      锁定属性                          : 0x00000000
      释放标志                          : 0x00000001
      锁定计数                          : 1
      挂起计数                          : 0
      锁定对象名                        : 4
      对象类型                          : 表
      表空间名称                        : USERSPACE1
      表模式                            : RIIHI
      表名                              : DEPARTMENT
      方式                              : IS - 意向共享

      锁定名称                          : 0x02000300000000000000000054
      锁定属性                          : 0x00000000
      释放标志                          : 0x40000000
      锁定计数                          : 1
      挂起计数                          : 0
      锁定对象名                        : 3
      对象类型                          : 表
      表空间名称                        : USERSPACE1
      表模式                            : RIIHI
      表名                              : STAFF
      方式                              : IX - 意向互斥

  挂起的锁定数: 6
  列表中的锁定数: 6
```

结尾:

结尾信息将在数据库释放时（最后一个应用程序断开连接后）生成:

```
8) 表事件 ...
  表模式: SYSIBM
  表名: SYSBOOT

  记录是清空的结果: FALSE
  表类型: 目录
  数据对象页数: 1
  读取的行数: 1
  写入的行数: 0
  溢出访问数: 0
  页重组数: 0
  表空间标识: 0
  表事件时间戳记: 11/25/2003 13:09:23.101214

9) 表事件 ...
  表模式: SYSIBM
  表名: SYSTABLES

  记录是清空的结果: FALSE
  表类型: 目录
  记录是清空的结果: FALSE
  表类型: 目录
  数据对象页数: 28
  索引对象页数: 17
  LOB 对象页数: 256
  读取的行数: 2
  写入的行数: 0
  溢出访问数: 0
  页重组数: 0
  表空间标识: 0
  表事件时间戳记: 11/25/2003 13:09:23.101265

10) 表事件 ...
```

```
  表模式: SYSIBM
  表名: SYSPLAN

  记录是清空的结果: FALSE
  表类型: 目录
  数据对象页数: 9
  索引对象页数: 5
  LOB 对象页数: 320
  读取的行数: 1
  写入的行数: 0
  溢出访问数: 0
  写入的行数: 0
  溢出访问数: 0
  页重组数: 0
  表空间标识: 0
  表事件时间戳记: 11/25/2003 13:09:23.101303

11) 表事件 ...
  表模式: SYSIBM
  表名: SYSDBAUTH

  记录是清空的结果: FALSE
  表类型: 目录
  数据对象页数: 1
  索引对象页数: 3
  读取的行数: 3
  写入的行数: 0
  溢出访问数: 0
  页重组数: 0
  表空间标识: 0
  表事件时间戳记: 11/25/2003 13:09:23.101337

12) 表事件 ...
  表模式: SYSIBM
  表名: SYSEVENTMONITORS
  表模式: SYSIBM
  表名: SYSEVENTMONITORS

  记录是清空的结果: FALSE
  表类型: 目录
  数据对象页数: 1
  索引对象页数: 3
  LOB 对象页数: 64
  读取的行数: 3
  写入的行数: 0
  溢出访问数: 0
  页重组数: 0
  表空间标识: 0
  表事件时间戳记: 11/25/2003 13:09:23.101382

13) 表事件 ...
  表模式: SYSIBM
  表名: SYSTABLESPACES

  记录是清空的结果: FALSE
  表类型: 目录
  数据对象页数: 1
  索引对象页数: 7
  读取的行数: 3
  索引对象页数: 7
  读取的行数: 3
  写入的行数: 0
  溢出访问数: 0
  页重组数: 0
  表空间标识: 0
  表事件时间戳记: 11/25/2003 13:09:23.101412
```

```
14) 表事件 ...
  表模式: SYSIBM
  表名: SYSBUFFERPOOLS

  记录是清空的结果: FALSE
  表类型: 目录
  数据对象页数: 1
  索引对象页数: 4
  读取的行数: 1
  写入的行数: 0
  溢出访问数: 0
  页重组数: 0
  表空间标识: 0
  表事件时间戳记: 11/25/2003 13:09:23.101452

15) 表事件 ...
  表模式: SYSIBM
  表名: SYSVERSIONS

  记录是清空的结果: FALSE
  表类型: 目录
  数据对象页数: 1
  索引对象页数: 3
  读取的行数: 1
  写入的行数: 0
  溢出访问数: 0
  页重组数: 0
  表空间标识: 0
  表事件时间戳记: 11/25/2003 13:09:23.101541

16) 表事件 ...
  表模式: RIIHI
  表名: STAFF

  记录是清空的结果: FALSE
  表类型: 用户
  数据对象页数: 1
  读取的行数: 36
  数据对象页数: 1
  读取的行数: 36
  写入的行数: 1
  溢出访问数: 0
  页重组数: 0
  表空间标识: 2
  表事件时间戳记: 11/25/2003 13:09:23.101890

17) 表事件 ...
  表模式: RIIHI
  表名: DEPARTMENT

  记录是清空的结果: FALSE
  表类型: 用户
  数据对象页数: 1
  读取的行数: 9
  写入的行数: 1
  溢出访问数: 0
  页重组数: 0
  表空间标识: 2
  表事件时间戳记: 11/25/2003 13:09:23.101918
```

相关概念:

- 第 55 页的『事件监视器』
- 第 124 页的『事件类型至逻辑数据组的映射』

相关任务:

• 第 57 页的『收集关于数据库系统事件的信息』

相关参考:

• 第 56 页的『事件类型』

事件记录及其相应的应用程序

在带有几百个相连的应用程序的活动数据库的事件跟踪中，跟踪与特定应用程序相关联的事件记录可能会非常麻烦。为提高可跟踪性，每个事件记录包括应用程序句柄和应用程序标识。它们允许您将每个记录与对其生成事件记录的应用程序相关联。

在应用程序有效期间，应用程序句柄（**agent_id**）在系统范围内是唯一的。但是，最终它将被重复使用（将使用 16 位计数器在分区数据库系统上生成此标识，它由协调分区号和 16 位计数器组成）。在大多数情况下，重复使用不存在问题，原因是从跟踪读取记录的应用程序能够检测到已终止的连接。例如，在跟踪中遇到带有已知 agent_ID 的连接头暗示带有此 agent_ID 的先前连接已终止。

应用程序标识是一个字符串标识，它包括时间戳记，并且承诺即使在停止并重新启动数据库管理器之后也一直是唯一的。

通过使用写至表事件监视器，查找特定应用程序的事件记录变得特别容易。在事件监视器表中，每行对应一个事件记录，应用程序句柄和应用程序标识是缺省列值。要查找给定应用程序的所有事件记录，只要对与特定应用程序标识对应的所有事件记录发出 SQL SELECT 语句就可以了。

相关概念:

• 第 85 页的『事件监视器自描述数据流』
• 第 55 页的『事件监视器』

相关任务:

• 第 57 页的『收集关于数据库系统事件的信息』

相关参考:

• 第 75 页的『事件监视器样本输出』

事件监视器自描述数据流

事件监视器的输出是一个逻辑数据分组二进制流，这些分组对于管道和文件事件监视器是完全一样的。可通过使用 db2evmon 命令或开发客户机应用程序来格式化数据流。此数据流显示为自描述格式。第 86 页的图 3 显示数据流的结构，而第 87 页的表 12 提供可返回的逻辑数据组和监视元素的一些示例。

注: 在示例和表中，将对标识使用描述性名称。在实际数据流中，将在这些名称前加上 **SQLM_ELM_** 前缀。例如，db_event 在事件监视器输出中显示为 SQLM_ELM_DB_EVENT。在实际数据流中，将在类型前加上 **SQLM_TYPE_** 前缀。例如，HEADER 在数据流中将显示为 SQLM_TYPE_HEADER。

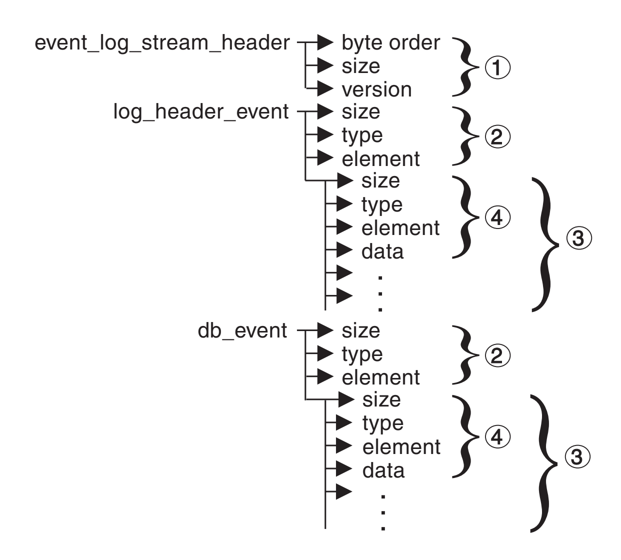


图 3. 事件监视器数据流

1. sqlm_event_log_data_stream_header 的结构与数据流中的其他头不同。版本字段确定是否能作为版本 8 数据流来处理输出。

   此头的大小和类型与版本 6 之前的事件监视器流的大小和类型相同。这允许应用程序确定事件监视器输出是自描述格式还是版本 6 之前的静态格式。

   注: 将通过从数据流读取 sizeof(sqlm_event_log_data_stream) 字节来抽取此监视元素。
2. 每个逻辑数据组以指示其大小和元素名称的头开始。因为大小元素包含哑元值以保留向后兼容性，所以这不适用于 event_log_stream_header。
3. 头中的大小元素指示该逻辑数据组中的所有数据的大小。
4. 监视元素信息在逻辑数据组头后面，并且也是自描述格式。

表 12. 样本事件数据流

逻辑数据组	数据流	描述
event_log_stream_header	sqlm_little_endian	未使用（因为与前发行版的兼容性）。
	200	未使用（因为与前发行版的兼容性）。
	sqlm_dbmon_version8	返回数据的数据库管理器的版本。只有版本 6、版本 7 和版本 8 的监视器才能以自描述格式写入数据。
log_header_event	100	逻辑数据组的大小。
	header	指示逻辑数据组的开头。
	log_header	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 － 32 位数字。
	byte_order	收集的监视元素的名称。
	little_endian	此元素的已收集值。
	2	此监视元素中存储的数据的大小。
	u16bit	监视元素类型 － 不带符号的 16 位数字。
	codepage_id	收集的监视元素的名称。
	850	此元素的已收集值。
db_event	100	逻辑数据组的大小。
	header	指示逻辑数据组的开头。
	db_event	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 － 不带符号的 32 位数字。
	lock_waits	收集的监视元素的名称。
	2	此元素的已收集值。

event_log_stream_header 标识返回数据的数据库管理器的版本。只有版本 6、版本 7 和版本 8 的监视器才能以自描述格式写入数据。如果使用上述任一版本的监视器，则可以开始处理自描述数据流。与快照监视器不同，事件监视器没有用来返回跟踪总大小的大小元素。event_log_stream_header 中显示的数字是为向后兼容性提供的哑元值。写入 event_log_stream_header 时，事件跟踪的总大小未知。通常可读取事件监视器跟踪直到文件或管道末尾。

日志头描述跟踪的特征，它包含各种信息，如在其中收集跟踪的服务器的内存模型（如小尾数法）和数据库的代码页。如果在其中读取跟踪的系统的内存模型与服务器的内存模型不同（例如，如果在 Windows 2000 系统上从 UNIX 服务器读取跟踪），则您可能必须对数字值执行字节交换。如果配置数据库时使用的语言与在其中读取跟踪的机器使用的语言不同，则可能还需要进行代码页转换。读取跟踪时，可使用大小元素来跳过跟踪中的逻辑数据组。

相关概念:

- 第 69 页的『事件监视器文件管理』
- 第 71 页的『事件监视器命名管道管理』
- 第 124 页的『事件类型至逻辑数据组的映射』

相关任务:

- 第 88 页的『在系统之间传送事件监视器数据』

相关参考:

- 第 75 页的『事件监视器样本输出』

相关样本:

- 『bldevm -- Builds the event monitor program, evm, on AIX (C)』
- 『bldevm.bat -- Builds event monitor program, evm, on Windows』
- 『evmprint.c -- Prints all events generated by an event monitor on AIX (C)』
- 『evmprint.c -- Prints all events generated by an event monitor on Windows (C)』
- 『evm.c -- Process event monitor data on AIX (C)』
- 『evm.c -- Process event monitor data on Windows (C)』
- 『evmread.c -- Read the event monitor self describing data stream on AIX (C)』
- 『evmread.c -- Read the event monitor self describing data stream on Windows (C)』

在系统之间传送事件监视器数据

对于在存储数字值时使用不同约定的系统，在系统间传送事件监视器信息时必须进行转换。UNIX 平台上的信息以小尾数法字节顺序存储，而 Windows 平台上的信息以大尾数法字节顺序存储。如果要在大尾数法平台上读取小尾数法源中的事件监视器数据（或反之），则需要进行字节转换。

过程:

要在逻辑数据组头和监视元素中转换数字值，请使用以下逻辑（用 C 显示）:

```
#include sqlmon.h
#define SWAP2(s) ((((s) >> 8) &0xFF) | (((s) << 8) &0xFF00))

#define SWAP4(l) ((((l) >> 24) &0xFF) | ((((l) &0xFF0000) >> 8) &0xFF00) \
                      | (((l) &0xFF00) << 8) | ((l) << 24))

#define SWAP8( where )                                                    \
 {                                                                        \
    sqluint32 temp;                                                       \
    temp = SWAP4(*(sqluint32 *) (where));                                 \
    * (sqluint32 *) (where) = SWAP4(* (((sqluint32 *) (where)) + 1));     \
    * (((sqluint32 *) (where)) + 1) = temp;                               \
 }

int HeaderByteReverse( sqlm_header_info * pHeader)
{   int rc = 0;

    pHeader->size = SWAP4(pHeader->size);
    pHeader->type = SWAP2(pHeader->type);
    pHeader->element = SWAP2(pHeader->element);

    return rc;
}

int DataByteReverse( char * dataBuf, sqluint32 dataSize)
{   int rc = 0;

    sqlm_header_info * pElemHeader = NULL;
    char * pElemData = NULL;
    sqluint32 dataOffset = 0;
    sqluint32 elemDataSize  = 0;
    sqluint32 elemHeaderSize = sizeof( sqlm_header_info);

    // For each of the elements in the datas tream that are numeric,
    // perform byte reversal.

    while( dataOffset < dataSize)
    {   /* byte reverse the element header */
```

```
        pElemHeader = (sqlm_header_info *)
                      ( dataBuf + dataOffset);

        rc = HeaderByteReverse( pElemHeader);
        if( rc != 0) return rc;
        // Remember the element data's size...it will be byte reversed
        // before we skip to the next element.
        elemDataSize = pElemHeader->size;

        /* byte reverse the element data */
        pElemData = (char *)
                      ( dataBuf + dataOffset + elemHeaderSize);

        if(pElemHeader->type == SQLM_TYPE_HEADER)
        {   rc = DataByteReverse( pElemData, pElemHeader->size);
            if( rc != 0) return rc;
        }
        else
 {   switch( pElemHeader->type)
            {   case SQLM_TYPE_16BIT:
                case SQLM_TYPE_U16BIT:
                    *(sqluint16 *)(pElemData) =
                                     SWAP2(*(short *)(pElemData));
                    break;
                case SQLM_TYPE_32BIT:
                case SQLM_TYPE_U32BIT:
                    *(sqluint32 *)(pElemData) =
                                     SWAP4(*(sqluint32 *)(pElemData));
                    break;
                case SQLM_TYPE_64BIT:
                case SQLM_TYPE_U64BIT:
                    SWAP8(pElemData);
                    break;
                default:
                    // Not a numeric type. Do nothing.
                    break;
            }
        }
        dataOffset = dataOffset + elemHeaderSize + elemDataSize;
    }

    return 0;
} /*  end of DataByteReverse */
```

相关概念:

- 第 69 页的『事件监视器文件管理』
- 第 71 页的『事件监视器命名管道管理』
- 第 85 页的『事件监视器自描述数据流』

第 2 部分 系统监视器参考

第 5 章 系统监视器逻辑数据组

快照监视器接口至逻辑数据组的映射

下表列示用来访问快照监视器数据的一些方法。所有快照监视器数据都存储在监视元素中，这些监视元素按逻辑数据组分类。每个 API 请求类型、CLP 命令和 SQL 管理视图仅从所有逻辑数据组的某个子集捕获监视器数据。

此表中列示的每个 API 请求类型、CLP 命令和 SQL 管理视图返回最右列中列示的逻辑数据组中的监视元素。

注:

1. 有一些 API 请求类型和 CLP 命令没有相应的 SQL 管理视图。对于其他 API 请求类型和 CLP 命令，各个 SQL 管理视图捕获关联逻辑数据组的子集。
2. 仅当关联监视开关设置为 ON 时，才返回某些监视元素。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

表 13. 快照监视器接口至逻辑数据组的映射

API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_APPLINFO_ALL	list applications [show detail]	应用程序	appl_info
SQLMA_DBASE_APPLINFO	list applications for database *dbname* [show detail]	应用程序	appl_info
SQLMA_DCS_APPLINFO_ALL	list dcs applications [show detail]		dcs_appl_info
SQLMA_DB2	get snapshot for dbm	SNAPDBM	db2
		SNAPFCM	fcm
		SNAPFCMPART	fcm_node
		SNAPUTIL	utility_info
		SNAPUTIL_PROGRESS	progress 和 progress_info
		SNAPDBM_MEMORY_POOL	memory_pool
	get dbm monitor switches	SNAPSWITCHES	switch_list
SQLMA_DBASE	get snapshot for database on *dbname*	SNAPDB	dbase
		SNAPDETAILLOG	detail_log
		SNAPSTORAGE_PATHS	db_storage_group
			rollforward
		SNAPTBSP	tablespace
		SNAPDB_MEMORY_POOL	memory_pool
SQLMA_DBASE_ALL	get snapshot for all databases	SNAPDB	dbase
		SNAPSTORAGE_PATHS	db_storage_group
			rollforward
		SNAPTBSP	tablespace
		SNAPDB_MEMORY_POOL	memory_pool
	list active databases		dbase

表 13. 快照监视器接口至逻辑数据组的映射 (续)

API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_DCS_DBASE	get snapshot for dcs database on *dbname*		dcs_dbase 和 stmt_transmissions
SQLMA_DCS_DBASE_ALL	get snapshot for all dcs databases		dcs_dbase 和 stmt_transmissions
SQLMA_DBASE_REMOTE	get snapshot for remote database on *dbname*		dbase_remote
SQLMA_DBASE_REMOTE_ALL	get snapshot for all remote databases		dbase_remote
SQLMA_APPL	get snapshot for application applid *appl-id*	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_AGENT_ID	get snapshot for application agentid *appl-handle*	SNAPAGENT	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_DBASE_APPLS	get snapshot for applications on *dbname*	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_APPL_ALL	get snapshot for all applications	SNAPAPPL	appl
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTATEMENT	stmt
		SNAPAGENT	agent
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_DCS_APPL	get snapshot for dcs application applid *appl-id*		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DCS_APPL_ALL	get snapshot for all dcs applications		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DCS_APPL_HANDLE	get snapshot for dcs application agentid *appl-handle*		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DCS_DBASE_APPLS	get snapshot for dcs applications on *dbname*		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions

表 13. 快照监视器接口至逻辑数据组的映射 (续)

API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_DBASE_APPLS_REMOTE	get snapshot for remote applications on *dbname*		dbase_appl
SQLMA_APPL_REMOTE_ALL	get snapshot for all remote applications		dbase_appl
SQLMA_DBASE_TABLES	get snapshot for tables on *dbname*	SNAPTAB	table
		SNAPTAB_REORG	table_reorg
			table_list
SQLMA_APPL_LOCKS	get snapshot for locks for application applid *appl-id*	SNAPLOCK、SNAPAPPL 和 SNAPLOCKWAIT	appl_lock_list、lock_wait 和 lock
SQLMA_APPL_LOCKS_AGENT_ID	get snapshot for locks for application agentid *appl-handle*	SNAPLOCK、SNAPAPPL 和 SNAPLOCKWAIT	appl_lock_list、lock_wait 和 lock
SQLMA_DBASE_LOCKS	get snapshot for locks on *dbname*	SNAPLOCK	appl_lock_list 和 lock
		SNAPLOCK 和 SNAPLOCKWAIT	db_lock_list 和 lock_wait
SQLMA_DBASE_TABLESPACES	get snapshot for tablespaces on *dbname*	SNAPTBSP	tablespace
		SNAPTBSPPART	tablespace 和 tablespace_nodeinfo
		SNAPTBSP_QUIESCER	tablespace_quiescer 和 tablespace_nodeinfo
		SNAPCONTAINER	tablespace_container 和 tablespace_nodeinfo
		SNAPTBSP_RANGE	tablespace_ranges 和 tablespace_nodeinfo
			tablespace_list 和 tablespace_nodeinfo
SQLMA_BUFFERPOOLS_ALL	get snapshot for all bufferpools	SNAPBP	bufferpool
SQLMA_DBASE_BUFFERPOOLS	get snapshot for bufferpools on *dbname*	SNAPBP	bufferpool
SQLMA_DYNAMIC_SQL	get snapshot for dynamic sql on *dbname*	SNAPDYN_SQL	dynsql
			dynsql_list

下图显示逻辑数据分组在快照数据流中可能出现的顺序。

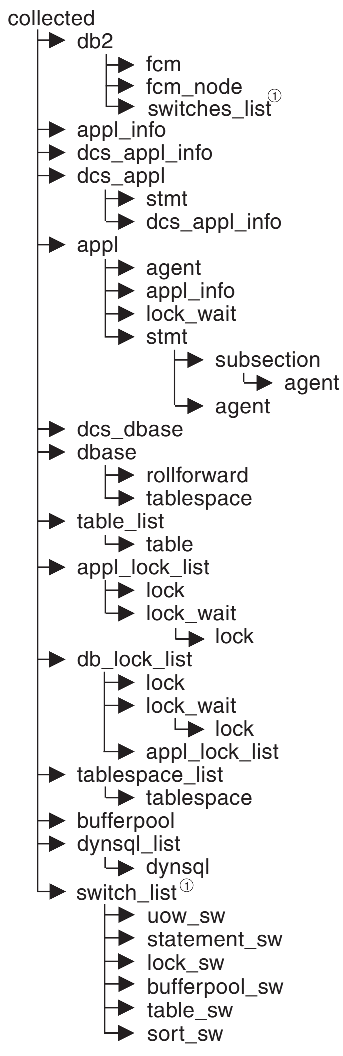


图 4. 数据流层次结构

注: 时间可能作为任何逻辑数据分组的一部分返回。

相关参考:

- 『Supported administrative SQL routines and views』（*Administrative SQL Routines and Views*）
- 『GET SNAPSHOT command』（*Command Reference*）

快照监视器逻辑数据组和监视元素

下表列示快照监视可能返回的逻辑数据分组和监视元素。

表 14. 快照监视器逻辑数据组和监视元素

快照逻辑数据组	监视元素
agent	第 182 页的『agent_pid – 进程或线程标识』 第 305 页的『lock_timeout_val – 锁定超时』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
appl	第 352 页的『acc_curs_blk － 接受的块游标请求数』
	第 392 页的『agent_sys_cpu_time － 代理程序使用的系统 CPU 时间』
	第 391 页的『agent_usr_cpu_time － 代理程序使用的用户 CPU 时间』
	第 191 页的『agents_stolen － 失窃代理程序数』
	第 176 页的『appl_con_time － 连接请求启动时间戳记』
	第 180 页的『appl_idle_time － 应用程序空闲时间』
	第 173 页的『appl_priority － 应用程序代理程序优先级』
	第 174 页的『appl_priority_type － 应用程序优先级类型』
	第 192 页的『associated_agents_top － 最大关联代理程序数』
	第 174 页的『authority_lvl － 用户权限级别』
	第 363 页的『binds_precompiles － 尝试的绑定次数／预编译次数』
	第 261 页的『cat_cache_inserts － 目录高速缓存插入数』
	第 260 页的『cat_cache_lookups － 目录高速缓存查询数』
	第 261 页的『cat_cache_overflows － 目录高速缓存溢出数』
	第 356 页的『commit_sql_stmts － 尝试的落实语句数』
	第 177 页的『conn_complete_time － 连接请求完成时间戳记』
	第 359 页的『ddl_sql_stmts － 数据定义语言（DDL）SQL 语句数』
	第 288 页的『deadlocks － 检测到的死锁数』
	第 257 页的『direct_read_reqs － 直接读取请求数』
	第 258 页的『direct_read_time － 直接读取时间』
	第 255 页的『direct_reads － 直接从数据库进行读取的读取操作数』
	第 257 页的『direct_write_reqs － 直接写入请求数』
	第 259 页的『direct_write_time － 直接写入时间』
	第 256 页的『direct_writes － 直接写入数据库的写入操作数』
	第 354 页的『dynamic_sql_stmts － 尝试的动态 SQL 语句数』
	第 355 页的『failed_sql_stmts － 失败的语句操作数』
	第 208 页的『hash_join_overflows － 散列连接溢出数』
	第 209 页的『hash_join_small_overflows － 散列连接小溢出数』
	第 424 页的『inbound_comm_address － 入站通信地址』
	第 359 页的『int_auto_rebinds － 内部自动重新绑定次数』
	第 360 页的『int_commits － 内部落实数』
	第 362 页的『int_deadlock_rollbacks － 死锁导致的内部回滚数』
	第 361 页的『int_rollbacks － 内部回滚数』
	第 339 页的『int_rows_deleted － 删除的内部行数』
	第 341 页的『int_rows_inserted － 插入的内部行数』
	第 340 页的『int_rows_updated － 更新的内部行数』
	第 397 页的『last_reset － 最后复位时间戳记』
	第 295 页的『lock_escalation － 锁定升级』
	第 294 页的『lock_timeouts － 锁定超时次数』
	第 305 页的『lock_timeout_val － 锁定超时』
	第 303 页的『lock_wait_time － 等待锁定的时间』
	第 302 页的『lock_waits － 锁定等待数』
	第 287 页的『locks_held － 挂起的锁定数』
	第 303 页的『locks_waiting － 等待锁定的当前代理程序数』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
appl（续）	第 390 页的『num_agents – 正在处理语句的代理程序数』
	第 352 页的『open_loc_curs – 打开的本地游标数』
	第 353 页的『open_loc_curs_blk – 打开的本地分块游标数』
	第 350 页的『open_rem_curs – 打开的远程游标数』
	第 350 页的『open_rem_curs_blk – 打开的远程分块游标数』
	第 265 页的『pkg_cache_inserts – 程序包高速缓存插入数』
	第 263 页的『pkg_cache_lookups – 程序包高速缓存查询数』
	第 222 页的『pool_data_l_reads – 缓冲池数据页逻辑读取数』
	第 224 页的『pool_data_p_reads – 缓冲池数据页物理读取数』
	第 226 页的『pool_data_writes – 缓冲池数据页写入数』
	第 227 页的『pool_index_l_reads – 缓冲池索引逻辑读取数』
	第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』
	第 231 页的『pool_index_writes – 缓冲池索引写入数』
	第 232 页的『pool_read_time – 缓冲池物理读总时间』
	第 224 页的『pool_temp_data_l_reads – 缓冲池临时数据逻辑读取数』
	第 225 页的『pool_temp_data_p_reads – 缓冲池临时数据物理读取数』
	第 228 页的『pool_temp_index_l_reads – 缓冲池临时索引逻辑读取数』
	第 230 页的『pool_temp_index_p_reads – 缓冲池临时索引物理读取数』
	第 252 页的『pool_temp_xda_l_reads – 逻辑读取的缓冲池临时 XDA 数据页数』
	第 253 页的『pool_temp_xda_p_reads – 物理读取的缓冲池临时 XDA 数据页数』
	第 233 页的『pool_write_time – 缓冲池物理写入总时间』
	第 249 页的『pool_xda_l_reads – 逻辑读取的缓冲池 XDA 数据页数』
	第 250 页的『pool_xda_p_reads – 物理读取的缓冲池 XDA 数据页数』
	第 251 页的『pool_xda_writes – 缓冲池 XDA 数据写入数』
	第 244 页的『prefetch_wait_time – 等待预取的时间』
	第 177 页的『prev_uow_stop_time – 上一个工作单元完成时间戳记』
	第 271 页的『priv_workspace_num_overflows – 专用工作空间溢出数』
	第 272 页的『priv_workspace_section_inserts – 专用工作空间段插入数』
	第 272 页的『priv_workspace_section_lookups – 专用工作空间段查询数』
	第 271 页的『priv_workspace_size_top – 最大专用工作空间大小』
	第 351 页的『rej_curs_blk – 拒绝的块游标请求数』
	第 357 页的『rollback_sql_stmts – 尝试的回滚语句数』
	第 335 页的『rows_deleted – 删除的行数』
	第 335 页的『rows_inserted – 插入的行数』
	第 338 页的『rows_read – 读取的行数』
	第 337 页的『rows_selected – 选择的行数』
	第 336 页的『rows_updated – 更新的行数』
	第 337 页的『rows_written – 写入的行数』
	第 357 页的『select_sql_stmts – 执行的 SELECT SQL 语句数』
	第 269 页的『shr_workspace_num_overflows – 共享工作空间溢出数』
	第 270 页的『shr_workspace_section_inserts – 共享工作空间段插入数』
	第 269 页的『shr_workspace_section_lookups – 共享工作空间段查询数』
	第 268 页的『shr_workspace_size_top – 最大共享工作空间大小』
	第 203 页的『sort_overflows – 排序溢出数』
	第 363 页的『sql_reqs_since_commit – 上次落实后的 SQL 请求数』
	第 354 页的『static_sql_stmts – 尝试的静态 SQL 语句数』
	第 364 页的『xquery_stmts – 尝试的 XQuery 语句数』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
appl（续）	第 206 页的『total_hash_joins – 散列连接总数』 第 208 页的『total_hash_loops – 总散列循环数』 第 202 页的『total_sort_time – 总排序时间』 第 201 页的『total_sorts – 总排序数』 第 358 页的『uid_sql_stmts – 执行的 Update/Insert/Delete SQL 语句数』 第 244 页的『unread_prefetch_pages – 未读取的预取页数』 第 180 页的『uow_comp_status – 工作单元完成状态』 第 179 页的『uow_elapsed_time – 最新工作单元耗用时间』 第 304 页的『uow_lock_wait_time – 工作单元等待锁定的总时间』 第 277 页的『uow_log_space_used – 使用的工作单元日志空间』 第 178 页的『uow_start_time – 工作单元开始时间戳记』 第 179 页的『uow_stop_time – 工作单元停止时间戳记』 第 290 页的『x_lock_escals – 互斥锁定升级数』
appl_id_info	第 158 页的『agent_id – 应用程序句柄（代理程序标识）』 第 163 页的『appl_id – 应用程序标识』 第 163 页的『appl_name – 应用程序名称』 第 159 页的『appl_status – 应用程序状态』 第 166 页的『auth_id – 授权标识』 第 168 页的『client_db_alias – 应用程序使用的数据库别名』 第 167 页的『client_nname – 客户机的配置 NNAME 』 第 168 页的『client_prdid – 客户机产品／版本标识』 第 161 页的『codepage_id – 应用程序使用的代码页的标识』 第 149 页的『db_name – 数据库名称』 第 150 页的『db_path – 数据库路径』 第 398 页的『input_db_alias – 输入数据库别名』 第 166 页的『sequence_no – 序号』 第 167 页的『session_auth_id – 会话授权标识』 第 162 页的『status_change_time – 应用程序状态更改时间』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
appl_info	第 158 页的『agent_id － 应用程序句柄（代理程序标识）』
	第 163 页的『appl_id － 应用程序标识』
	第 163 页的『appl_name － 应用程序名称』
	第 267 页的『appl_section_inserts － 段插入数监视元素』
	第 266 页的『appl_section_lookups － 段查询』
	第 159 页的『appl_status － 应用程序状态』
	第 166 页的『auth_id － 授权标识』
	第 174 页的『authority_lvl － 用户权限级别』
	第 168 页的『client_db_alias － 应用程序使用的数据库别名』
	第 167 页的『client_nname － 客户机的配置 NNAME 』
	第 171 页的『client_pid － 客户机进程标识』
	第 172 页的『client_platform － 客户机操作平台』
	第 168 页的『client_prdid － 客户机产品／版本标识』
	第 172 页的『client_protocol － 客户机通信协议』
	第 161 页的『codepage_id － 应用程序使用的代码页的标识』
	第 182 页的『coord_agent_pid － 协调程序代理程序』
	第 176 页的『coord_node － 协调节点』
	第 171 页的『corr_token － DRDA 关联标记』
	第 149 页的『db_name － 数据库名称』
	第 150 页的『db_path － 数据库路径』
	第 170 页的『execution_id － 用户登录标识』
	第 398 页的『input_db_alias － 输入数据库别名』
	第 193 页的『num_assoc_agents － 关联代理程序数』
	第 166 页的『sequence_no － 序号』
	第 162 页的『status_change_time － 应用程序状态更改时间』
	第 173 页的『territory_code － 数据库地域代码』
	第 448 页的『tpmon_acc_str － TP 监视器客户机记帐字符串』
	第 448 页的『tpmon_client_app － TP 监视器客户机应用程序名称』
	第 447 页的『tpmon_client_userid － TP 监视器客户机用户标识』
	第 447 页的『tpmon_client_wkstn － TP 监视器客户机工作站名称』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
appl_lock_list	第 158 页的『agent_id － 应用程序句柄（代理程序标识）』 第 163 页的『appl_id － 应用程序标识』 第 163 页的『appl_name － 应用程序名称』 第 159 页的『appl_status － 应用程序状态』 第 166 页的『auth_id － 授权标识』 第 168 页的『client_db_alias － 应用程序使用的数据库别名』 第 161 页的『codepage_id － 应用程序使用的代码页的标识』 第 287 页的『locks_held － 挂起的锁定数』 第 303 页的『locks_waiting － 等待锁定的当前代理程序数』 第 303 页的『lock_wait_time － 等待锁定的时间』 第 166 页的『sequence_no － 序号』 第 167 页的『session_auth_id － 会话授权标识』 第 162 页的『status_change_time － 应用程序状态更改时间』
appl_remote	第 356 页的『commit_sql_stmts － 尝试的落实语句数』 第 451 页的『create_nickname － 创建昵称数』 第 456 页的『create_nickname_time － 创建昵称响应时间』 第 449 页的『datasource_name － 数据源名称』 第 149 页的『db_name － 数据库名称』 第 451 页的『delete_sql_stmts － 删除数』 第 455 页的『delete_time － 删除响应时间』 第 355 页的『failed_sql_stmts － 失败的语句操作数』 第 450 页的『insert_sql_stmts － 插入数』 第 454 页的『insert_time － 插入响应时间』 第 456 页的『passthru_time － 传递时间』 第 452 页的『passthrus － 传递数』 第 457 页的『remote_lock_time － 远程锁定时间』 第 453 页的『remote_locks － 远程锁定数』 第 357 页的『rollback_sql_stmts － 尝试的回滚语句数』 第 335 页的『rows_deleted － 删除的行数』 第 335 页的『rows_inserted － 插入的行数』 第 337 页的『rows_selected － 选择的行数』 第 336 页的『rows_updated － 更新的行数』 第 357 页的『select_sql_stmts － 执行的 SELECT SQL 语句数』 第 454 页的『select_time － 查询响应时间』 第 453 页的『sp_rows_selected － 存储过程返回的行数』 第 456 页的『stored_proc_time － 存储过程时间』 第 452 页的『stored_procs － 存储过程数』 第 450 页的『update_sql_stmts － 更新数』 第 455 页的『update_time － 更新响应时间』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
bufferpool	第 245 页的『block_ios － 块 IO 请求数』 第 243 页的『bp_name － 缓冲池名称』 第 222 页的『bp_id － 缓冲池标识』 第 149 页的『db_name － 数据库名称』 第 150 页的『db_path － 数据库路径』 第 257 页的『direct_read_reqs － 直接读取请求数』 第 255 页的『direct_reads － 直接从数据库进行读取的读取操作数』 第 258 页的『direct_read_time － 直接读取时间』 第 257 页的『direct_write_reqs － 直接写入请求数』 第 256 页的『direct_writes － 直接写入数据库的写入操作数』 第 259 页的『direct_write_time － 直接写入时间』 第 233 页的『files_closed － 关闭的数据库文件数』 第 398 页的『input_db_alias － 输入数据库别名』 第 246 页的『pages_from_block_ios － 块 IO 读取的总页数』 第 245 页的『pages_from_vectored_ios － 向量 IO 读取的总页数』 第 246 页的『physical_page_maps － 物理页映射数』 第 239 页的『pool_async_data_read_reqs － 缓冲池异步读取请求数』 第 234 页的『pool_async_data_reads － 缓冲池异步数据读取数』 第 235 页的『pool_async_data_writes － 缓冲池异步数据写入数』 第 239 页的『pool_async_index_read_reqs － 缓冲池异步索引读取请求数』 第 236 页的『pool_async_index_reads － 缓冲池异步索引读取数』 第 235 页的『pool_async_index_writes － 缓冲池异步索引写入数』 第 237 页的『pool_async_read_time － 缓冲池异步读取时间』 第 238 页的『pool_async_write_time － 缓冲池异步写入时间』 第 247 页的『pool_async_xda_read_reqs － 缓冲池异步 XDA 读取请求数』 第 247 页的『pool_async_xda_reads － 缓冲池异步 XDA 数据读取数』 第 248 页的『pool_async_xda_writes － 缓冲池异步 XDA 数据写入数』 第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』 第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』 第 226 页的『pool_data_writes － 缓冲池数据页写入数』 第 227 页的『pool_index_l_reads － 缓冲池索引逻辑读取数』 第 229 页的『pool_index_p_reads － 缓冲池索引物理读取数』 第 231 页的『pool_index_writes － 缓冲池索引写入数』 第 232 页的『pool_read_time － 缓冲池物理读总时间』 第 241 页的『pool_no_victim_buffer － 缓冲池没有牺牲缓冲区的次数』 第 224 页的『pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数』 第 225 页的『pool_temp_data_p_reads － 缓冲池临时数据物理读取数』 第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』 第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』 第 252 页的『pool_temp_xda_l_reads － 逻辑读取的缓冲池临时 XDA 数据页数』 第 253 页的『pool_temp_xda_p_reads － 物理读取的缓冲池临时 XDA 数据页数』 第 233 页的『pool_write_time － 缓冲池物理写入总时间』 第 249 页的『pool_xda_l_reads － 逻辑读取的缓冲池 XDA 数据页数』 第 250 页的『pool_xda_p_reads － 物理读取的缓冲池 XDA 数据页数』 第 251 页的『pool_xda_writes － 缓冲池 XDA 数据写入数』 第 244 页的『vectored_ios － 向量 IO 请求数』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
bufferpool_nodeinfo	第 254 页的『bp_cur_buffsz － 缓冲池的当前大小』 第 254 页的『bp_new_buffsz － 新的缓冲池大小』 第 254 页的『bp_pages_left_to_remove － 要除去的余下页数』 第 255 页的『bp_tbsp_use_count － 映射至缓冲池的表空间数』 第 175 页的『node_number － 节点号』
collected	第 175 页的『node_number － 节点号』 第 145 页的『server_db2_type － 受监视的（服务器）节点上的数据库管理器类型』 第 145 页的『server_instance_name － 服务器实例名称』 第 144 页的『server_nname － 监视（服务器）数据库分区上的配置 NNAME 』 第 146 页的『server_prdid － 服务器产品／版本标识』 第 146 页的『server_version － 服务器版本』 switch_list 监视开关控制数据 第 398 页的『time_stamp － 快照时间』 第 148 页的『time_zone_disp － 时区偏移』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
db2	第 191 页的『agents_created_empty_pool － 由于空的代理程序池而创建的代理程序数』
	第 190 页的『agents_from_pool － 从池中分配的代理程序数』
	第 188 页的『agents_registered － 注册的代理程序数』
	第 189 页的『agents_registered_top － 最大已注册代理程序数』
	第 191 页的『agents_stolen － 失窃代理程序数』
	第 188 页的『agents_waiting_on_token － 正在等待标记的代理程序数』
	第 189 页的『agents_waiting_top － 正在等待的代理程序的最大数目』
	第 192 页的『comm_private_mem － 已落实专用内存』
	第 186 页的『con_local_dbases － 带有当前连接的本地数据库数』
	第 191 页的『coord_agents_top － 最大协调代理程序数』
	第 144 页的『db2start_time － 启动数据库管理器时间戳记』
	第 152 页的『db_status － 数据库状态』
	第 418 页的『gw_total_cons － 对 DB2 Connect 尝试连接的总数』
	第 418 页的『gw_cur_cons － DB2 Connect 的当前连接数』
	第 419 页的『gw_cons_wait_host － 等待主机应答的连接数』
	第 419 页的『gw_cons_wait_client － 等待客户机发送请求的连接数』
	第 190 页的『idle_agents － 空闲代理程序数』
	第 397 页的『last_reset － 最后复位时间戳记』
	第 185 页的『local_cons － 本地连接数』
	第 185 页的『local_cons_in_exec － 在数据库管理器中执行的本地连接数』
	第 193 页的『max_agent_overflows － 最大代理程序溢出数』
	第 194 页的『num_gw_conn_switches － 连接交换次数』
	第 399 页的『num_nodes_in_db2_instance － 分区中的节点数』
	第 200 页的『piped_sorts_requested － 请求的管道排序数』
	第 200 页的『piped_sorts_accepted － 接受的管道排序数』
	第 207 页的『post_threshold_hash_joins － 散列连接阈值』
	第 199 页的『post_threshold_sorts － 超过阈值后的排序数』
	第 147 页的『product_name － 产品名称』
	第 184 页的『rem_cons_in － 与数据库管理器的远程连接数』
	第 184 页的『rem_cons_in_exec － 在数据库管理器中执行的远程连接数』
	第 147 页的『service_level － 服务级别』
	第 162 页的『smallest_log_avail_node － 带有最少可用日志空间的节点』
	第 199 页的『sort_heap_allocated － 分配的总排序堆』
	第 204 页的『sort_heap_top － 排序专用堆高水位标记』
	switch_list 监视开关控制数据
db_lock_list	第 187 页的『appls_cur_cons － 当前连接的应用程序数』
	第 149 页的『db_name － 数据库名称』
	第 150 页的『db_path － 数据库路径』
	第 398 页的『input_db_alias － 输入数据库别名』
	第 287 页的『locks_held － 挂起的锁定数』
	第 303 页的『locks_waiting － 等待锁定的当前代理程序数』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
dbase	第 206 页的『active_hash_joins － 活动散列连接数』 第 203 页的『active_sorts － 活动排序数』 第 390 页的『agents_top － 创建的代理程序数』 第 162 页的『appl_id_oldest_xact － 带有最旧事务的应用程序』 第 267 页的『appl_section_inserts － 段插入数监视元素』 第 266 页的『appl_section_lookups － 段查询』 第 187 页的『appls_cur_cons － 当前连接的应用程序数』 第 187 页的『appls_in_db2 － 当前在数据库中执行的应用程序数』 第 363 页的『binds_precompiles － 尝试的绑定次数／预编译次数』 第 261 页的『cat_cache_inserts － 目录高速缓存插入数』 第 260 页的『cat_cache_lookups － 目录高速缓存查询数』 第 261 页的『cat_cache_overflows － 目录高速缓存溢出数』 第 262 页的『cat_cache_size_top － 目录高速缓存高水位标记』 第 153 页的『catalog_node － 目录节点号』 第 152 页的『catalog_node_name － 目录节点网络名』 第 356 页的『commit_sql_stmts － 尝试的落实语句数』 第 176 页的『connections_top － 最大并行连接数』 第 191 页的『coord_agents_top － 最大协调代理程序数』 第 150 页的『db_conn_time － 数据库激活时间戳记』 第 273 页的『db_heap_top － 分配的最大数据库堆』 第 153 页的『db_location － 数据库位置』 第 149 页的『db_name － 数据库名称』 第 150 页的『db_path － 数据库路径』 第 152 页的『db_status － 数据库状态』 第 359 页的『ddl_sql_stmts － 数据定义语言（DDL）SQL 语句数』 第 288 页的『deadlocks － 检测到的死锁数』 第 257 页的『direct_read_reqs － 直接读取请求数』 第 258 页的『direct_read_time － 直接读取时间』 第 255 页的『direct_reads － 直接从数据库进行读取的读取操作数』 第 257 页的『direct_write_reqs － 直接写入请求数』 第 259 页的『direct_write_time － 直接写入时间』 第 256 页的『direct_writes － 直接写入数据库的写入操作数』 第 354 页的『dynamic_sql_stmts － 尝试的动态 SQL 语句数』 第 355 页的『failed_sql_stmts － 失败的语句操作数』 第 233 页的『files_closed － 关闭的数据库文件数』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
dbase（续）	第 208 页的『hash_join_overflows – 散列连接溢出数』
	第 209 页的『hash_join_small_overflows – 散列连接小溢出数』
	第 398 页的『input_db_alias – 输入数据库别名』
	第 359 页的『int_auto_rebinds – 内部自动重新绑定次数』
	第 360 页的『int_commits – 内部落实数』
	第 362 页的『int_deadlock_rollbacks – 死锁导致的内部回滚数』
	第 361 页的『int_rollbacks – 内部回滚数』
	第 339 页的『int_rows_deleted – 删除的内部行数』
	第 341 页的『int_rows_inserted – 插入的内部行数』
	第 340 页的『int_rows_updated – 更新的内部行数』
	第 153 页的『last_backup – 上次备份时间戳记』
	第 397 页的『last_reset – 最后复位时间戳记』
	第 289 页的『lock_escals – 锁定升级数』
	第 288 页的『lock_list_in_use – 使用中的锁定列表内存总量』
	第 294 页的『lock_timeouts – 锁定超时次数』
	第 303 页的『lock_wait_time – 等待锁定的时间』
	第 302 页的『lock_waits – 锁定等待数』
	第 287 页的『locks_held – 挂起的锁定数』
	第 303 页的『locks_waiting – 等待锁定的当前代理程序数』
	第 279 页的『log_held_by_dirty_pages – 脏页占用的日志空间量』
	第 281 页的『log_read_time – 日志读取时间』
	第 276 页的『log_reads – 读取的日志页数』
	第 280 页的『log_to_redo_for_recovery – 要为恢复重做的日志量』
	第 280 页的『log_write_time – 日志写入时间』
	第 277 页的『log_writes – 写入的日志页数』
	第 193 页的『num_assoc_agents – 关联代理程序数』
	第 154 页的『num_db_storage_paths – 自动存储器路径数』
	第 301 页的『num_indoubt_trans – 不确定事务数』
	第 283 页的『num_log_buffer_full – 变满的日志缓冲区的数目』
	第 283 页的『num_log_data_found_in_buffer – 在缓冲区中找到日志数据的次数』
	第 282 页的『num_log_part_page_io – 部分日志页写入数』
	第 282 页的『num_log_read_io – 日志读取数』
	第 281 页的『num_log_write_io – 日志写入次数』
	第 265 页的『pkg_cache_inserts – 程序包高速缓存插入数』
	第 263 页的『pkg_cache_lookups – 程序包高速缓存查询数』
	第 265 页的『pkg_cache_num_overflows – 程序包高速缓存溢出数』
	第 266 页的『pkg_cache_size_top – 程序包高速缓存高水位标记』
	第 239 页的『pool_async_data_read_reqs – 缓冲池异步读取请求数』
	第 234 页的『pool_async_data_reads – 缓冲池异步数据读取数』
	第 235 页的『pool_async_data_writes – 缓冲池异步数据写入数』
	第 239 页的『pool_async_index_read_reqs – 缓冲池异步索引读取请求数』
	第 236 页的『pool_async_index_reads – 缓冲池异步索引读取数』
	第 235 页的『pool_async_index_writes – 缓冲池异步索引写入数』
	第 237 页的『pool_async_read_time – 缓冲池异步读取时间』
	第 247 页的『pool_async_xda_read_reqs – 缓冲池异步 XDA 读取请求数』
	第 247 页的『pool_async_xda_reads – 缓冲池异步 XDA 数据读取数』
	第 248 页的『pool_async_xda_writes – 缓冲池异步 XDA 数据写入数』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
dbase（续）	第 238 页的『pool_async_write_time － 缓冲池异步写入时间』 第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』 第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』 第 226 页的『pool_data_writes － 缓冲池数据页写入数』 第 240 页的『pool_drty_pg_steal_clns － 触发的缓冲池牺牲页清除程序数』 第 242 页的『pool_drty_pg_thrsh_clns － 触发的缓冲池阈值清除程序数』 第 227 页的『pool_index_l_reads － 缓冲池索引逻辑读取数』 第 229 页的『pool_index_p_reads － 缓冲池索引物理读取数』 第 231 页的『pool_index_writes － 缓冲池索引写入数』 第 240 页的『pool_lsn_gap_clns － 触发的缓冲池日志空间清除程序数』 第 241 页的『pool_no_victim_buffer － 缓冲池没有牺牲缓冲区的次数』 第 232 页的『pool_read_time － 缓冲池物理读总时间』 第 224 页的『pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数』 第 225 页的『pool_temp_data_p_reads － 缓冲池临时数据物理读取数』 第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』 第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』 第 252 页的『pool_temp_xda_l_reads － 逻辑读取的缓冲池临时 XDA 数据页数』 第 253 页的『pool_temp_xda_p_reads － 物理读取的缓冲池临时 XDA 数据页数』 第 233 页的『pool_write_time － 缓冲池物理写入总时间』 第 249 页的『pool_xda_l_reads － 逻辑读取的缓冲池 XDA 数据页数』 第 250 页的『pool_xda_p_reads － 物理读取的缓冲池 XDA 数据页数』 第 251 页的『pool_xda_writes － 缓冲池 XDA 数据写入数』 第 207 页的『post_shrthreshold_hash_joins － 阈值后散列连接数』 第 205 页的『post_shrthreshold_sorts － 阈值后排序数』 第 271 页的『priv_workspace_num_overflows － 专用工作空间溢出数』 第 272 页的『priv_workspace_section_inserts － 专用工作空间段插入数』 第 272 页的『priv_workspace_section_lookups － 专用工作空间段查询数』 第 271 页的『priv_workspace_size_top － 最大专用工作空间大小』 第 357 页的『rollback_sql_stmts － 尝试的回滚语句数』 第 335 页的『rows_deleted － 删除的行数』 第 335 页的『rows_inserted － 插入的行数』 第 338 页的『rows_read － 读取的行数』 第 337 页的『rows_selected － 选择的行数』 第 336 页的『rows_updated － 更新的行数』 第 274 页的『sec_log_used_top － 使用的最大辅助日志空间』 第 276 页的『sec_logs_allocated － 目前分配的辅助日志数』 第 357 页的『select_sql_stmts － 执行的 SELECT SQL 语句数』 第 147 页的『server_platform － 服务器操作系统』 第 269 页的『shr_workspace_num_overflows － 共享工作空间溢出数』 第 270 页的『shr_workspace_section_inserts － 共享工作空间段插入数』 第 269 页的『shr_workspace_section_lookups － 共享工作空间段查询数』 第 268 页的『shr_workspace_size_top － 最大共享工作空间大小』 第 199 页的『sort_heap_allocated － 分配的总排序堆』 第 203 页的『sort_overflows － 排序溢出数』 第 204 页的『sort_shrheap_allocated － 当前分配的共享排序堆』 第 204 页的『sort_shrheap_top － 排序共享堆高水位标记』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
dbase（续）	第 354 页的『static_sql_stmts － 尝试的静态 SQL 语句数』
	第 275 页的『tot_log_used_top － 使用的最大总日志空间』
	第 186 页的『total_cons － 数据库激活以后的连接数』
	第 206 页的『total_hash_joins － 散列连接总数』
	第 208 页的『total_hash_loops － 总散列循环数』
	第 278 页的『total_log_available － 可用的总日志量』
	第 278 页的『total_log_used － 使用的总日志空间』
	第 193 页的『total_sec_cons － 辅助连接数』
	第 202 页的『total_sort_time － 总排序时间』
	第 201 页的『total_sorts － 总排序数』
	第 358 页的『uid_sql_stmts － 执行的 Update/Insert/Delete SQL 语句数』
	第 244 页的『unread_prefetch_pages － 未读取的预取页数』
	第 290 页的『x_lock_escals － 互斥锁定升级数』
	第 364 页的『xquery_stmts － 尝试的 XQuery 语句数』
dbase_remote	第 356 页的『commit_sql_stmts － 尝试的落实语句数』
	第 451 页的『create_nickname － 创建昵称数』
	第 456 页的『create_nickname_time － 创建昵称响应时间』
	第 449 页的『datasource_name － 数据源名称』
	第 149 页的『db_name － 数据库名称』
	第 451 页的『delete_sql_stmts － 删除数』
	第 455 页的『delete_time － 删除响应时间』
	第 450 页的『disconnects － 断开连接次数』
	第 355 页的『failed_sql_stmts － 失败的语句操作数』
	第 450 页的『insert_sql_stmts － 插入数』
	第 454 页的『insert_time － 插入响应时间』
	第 456 页的『passthru_time － 传递时间』
	第 452 页的『passthrus － 传递数』
	第 457 页的『remote_lock_time － 远程锁定时间』
	第 453 页的『remote_locks － 远程锁定数』
	第 357 页的『rollback_sql_stmts － 尝试的回滚语句数』
	第 335 页的『rows_deleted － 删除的行数』
	第 335 页的『rows_inserted － 插入的行数』
	第 337 页的『rows_selected － 选择的行数』
	第 336 页的『rows_updated － 更新的行数』
	第 357 页的『select_sql_stmts － 执行的 SELECT SQL 语句数』
	第 454 页的『select_time － 查询响应时间』
	第 453 页的『sp_rows_selected － 存储过程返回的行数』
	第 456 页的『stored_proc_time － 存储过程时间』
	第 452 页的『stored_procs － 存储过程数』
	第 186 页的『total_cons － 数据库激活以后的连接数』
	第 450 页的『update_sql_stmts － 更新数』
	第 455 页的『update_time － 更新响应时间』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
db_storage_group	第 154 页的『db_storage_path – 自动存储器路径』 第 154 页的『sto_path_free_sz – 自动存储器路径可用空间』 第 155 页的『fs_used_size – 文件系统上的已用空间量』 第 155 页的『fs_total_size – 文件系统总大小』 第 156 页的『fs_id – 唯一文件系统标识号』 第 157 页的『fs_type – 文件系统类型』 第 175 页的『node_number – 节点号』
dcs_appl	第 180 页的『appl_idle_time – 应用程序空闲时间』 第 356 页的『commit_sql_stmts – 尝试的落实语句数』 第 442 页的『elapsed_exec_time – 语句执行耗用时间』 第 355 页的『failed_sql_stmts – 失败的语句操作数』 第 417 页的『gw_con_time – DB2 Connect 网关首次启动的连接』 第 419 页的『gw_exec_time – DB2 Connect 网关处理所耗用的时间』 第 443 页的『host_response_time – 主机响应时间』 第 424 页的『inbound_bytes_received – 接收的入站字节数』 第 426 页的『inbound_bytes_sent – 发送的入站字节数』 第 397 页的『last_reset – 最后复位时间戳记』 第 428 页的『max_data_received_128 – 接收的出站字节数在 1 到 128 字节之间的语句数』 第 429 页的『max_data_received_256 – 接收的出站字节数在 129 到 256 字节之间的语句数』 第 430 页的『max_data_received_512 – 接收的出站字节数在 257 到 512 字节之间的语句数』 第 431 页的『max_data_received_1024 – 接收的出站字节数在 513 到 1024 字节之间的语句数』 第 432 页的『max_data_received_2048 – 发送的出站字节数在 1025 到 2048 字节之间的语句数』 第 433 页的『max_data_received_4096 – 接收的出站字节数在 2049 到 4096 字节之间的语句数』 第 434 页的『max_data_received_8192 – 接收的出站字节数在 4097 到 8192 字节之间的语句数』 第 435 页的『max_data_received_16384 – 发送的出站字节数在 8193 到 16384 字节之间的语句数』 第 436 页的『max_data_received_31999 – 接收的出站字节数在 16385 到 31999 字节之间的语句数』 第 437 页的『max_data_received_64000 – 接收的出站字节数在 32000 到 64000 字节之间的语句数』 第 438 页的『max_data_received_gt64000 – 接收的出站字节数高于 64000 的语句数』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
dcs_appl（续）	第 428 页的『max_data_sent_128 － 发送的出站字节数在 1 到 128 字节之间的语句数』 第 429 页的『max_data_sent_256 － 发送的出站字节数在 129 到 256 字节之间的语句数』 第 430 页的『max_data_sent_512 － 发送的出站字节数在 257 到 512 字节之间的语句数』 第 431 页的『max_data_sent_1024 － 发送的出站字节数在 513 到 1024 字节之间的语句数』 第 432 页的『max_data_sent_2048 － 发送的出站字节数在 1025 到 2048 字节之间的语句数』 第 433 页的『max_data_sent_4096 － 发送的出站字节数在 2049 到 4096 字节之间的语句数』 第 433 页的『max_data_sent_8192 － 发送的出站字节数在 4097 到 8192 字节之间的语句数』 第 434 页的『max_data_sent_16384 － 发送的出站字节数在 8193 到 16384 字节之间的语句数』 第 435 页的『max_data_sent_31999 － 发送的出站字节数在 16385 到 31999 字节之间的语句数』 第 436 页的『max_data_sent_64000 － 发送的出站字节数在 32000 到 64000 字节之间的语句数』 第 437 页的『max_data_sent_gt64000 － 发送的出站字节数高于 64000 的语句数』 第 438 页的『max_network_time_1_ms － 网络时间最多为 1 ms 的语句数』 第 439 页的『max_network_time_4_ms － 网络时间在 1 到 4 ms 之间的语句数』 第 439 页的『max_network_time_16_ms － 网络时间在 4 到 16 ms 之间的语句数』 第 440 页的『max_network_time_100_ms － 网络时间在 16 到 100 ms 之间的语句数』 第 440 页的『max_network_time_500_ms － 网络时间在 100 到 500 ms 之间的语句数』 第 441 页的『max_network_time_gt500_ms － 网络时间大于 500 ms 的语句数』 第 441 页的『network_time_top － 语句的最长网络时间』 第 442 页的『network_time_bottom － 语句的最短网络时间』 第 421 页的『open_cursors － 打开的游标数』 第 425 页的『outbound_bytes_received － 接收的出站字节数』 第 425 页的『outbound_bytes_sent － 发送的出站字节数』 第 177 页的『prev_uow_stop_time － 上一个工作单元完成时间戳记』 第 357 页的『rollback_sql_stmts － 尝试的回滚语句数』 第 337 页的『rows_selected － 选择的行数』 第 420 页的『sql_stmts － 尝试的 SQL 语句数』 第 448 页的『tpmon_acc_str － TP 监视器客户机记帐字符串』 第 448 页的『tpmon_client_app － TP 监视器客户机应用程序名称』 第 447 页的『tpmon_client_userid － TP 监视器客户机用户标识』 第 447 页的『tpmon_client_wkstn － TP 监视器客户机工作站名称』 第 180 页的『uow_comp_status － 工作单元完成状态』 第 179 页的『uow_elapsed_time － 最新工作单元耗用时间』 第 178 页的『uow_start_time － 工作单元开始时间戳记』 第 179 页的『uow_stop_time － 工作单元停止时间戳记』 第 442 页的『xid － 事务标识』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
dcs_appl_info	第 158 页的『agent_id － 应用程序句柄（代理程序标识）』 第 422 页的『agent_status － DCS 应用程序代理程序数』 第 163 页的『appl_id － 应用程序标识』 第 163 页的『appl_name － 应用程序名称』 第 166 页的『auth_id － 授权标识』 第 167 页的『client_nname － 客户机的配置 NNAME 』 第 171 页的『client_pid － 客户机进程标识』 第 172 页的『client_platform － 客户机操作平台』 第 168 页的『client_prdid － 客户机产品／版本标识』 第 172 页的『client_protocol － 客户机通信协议』 第 161 页的『codepage_id － 应用程序使用的代码页的标识』 第 422 页的『dcs_appl_status － DCS 应用程序状态』 第 416 页的『dcs_db_name － DCS 数据库名称』 第 170 页的『execution_id － 用户登录标识』 第 417 页的『gw_db_alias － 网关上的数据库别名』 第 423 页的『host_ccsid － 主机编码字符集标识』 第 417 页的『host_db_name － 主机数据库名称』 第 169 页的『host_prdid － 主机产品／版本标识』 第 424 页的『inbound_comm_address － 入站通信地址』 第 169 页的『outbound_appl_id － 出站应用程序标识』 第 424 页的『outbound_comm_address － 出站通信地址』 第 423 页的『outbound_comm_protocol － 出站通信协议』 第 170 页的『outbound_sequence_no － 出站序号』 第 166 页的『sequence_no － 序号』 第 162 页的『status_change_time － 应用程序状态更改时间』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
dcs_dbase	第 356 页的『commit_sql_stmts － 尝试的落实语句数』
	第 445 页的『con_elapsed_time － 最新连接耗用时间』
	第 445 页的『con_response_time － 连接的最新响应时间』
	第 416 页的『dcs_db_name － DCS 数据库名称』
	第 442 页的『elapsed_exec_time － 语句执行耗用时间』
	第 355 页的『failed_sql_stmts － 失败的语句操作数』
	第 446 页的『gw_comm_error_time － 通信错误时间』
	第 445 页的『gw_comm_errors － 通信错误数』
	第 417 页的『gw_con_time － DB2 Connect 网关首次启动的连接』
	第 418 页的『gw_connections_top － 与主机数据库的最大并行连接数』
	第 419 页的『gw_cons_wait_client － 等待客户机发送请求的连接数』
	第 419 页的『gw_cons_wait_host － 等待主机应答的连接数』
	第 418 页的『gw_cur_cons － DB2 Connect 的当前连接数』
	第 418 页的『gw_total_cons － 对 DB2 Connect 尝试连接的总数』
	第 417 页的『host_db_name － 主机数据库名称』
	第 443 页的『host_response_time － 主机响应时间』
	第 424 页的『inbound_bytes_received － 接收的入站字节数』
	第 397 页的『last_reset － 最后复位时间戳记』
	第 428 页的『max_data_received_128 － 接收的出站字节数在 1 到 128 字节之间的语句数』
	第 429 页的『max_data_received_256 － 接收的出站字节数在 129 到 256 字节之间的语句数』
	第 430 页的『max_data_received_512 － 接收的出站字节数在 257 到 512 字节之间的语句数』
	第 431 页的『max_data_received_1024 － 接收的出站字节数在 513 到 1024 字节之间的语句数』
	第 432 页的『max_data_received_2048 － 发送的出站字节数在 1025 到 2048 字节之间的语句数』
	第 433 页的『max_data_received_4096 － 接收的出站字节数在 2049 到 4096 字节之间的语句数』
	第 434 页的『max_data_received_8192 － 接收的出站字节数在 4097 到 8192 字节之间的语句数』
	第 435 页的『max_data_received_16384 － 发送的出站字节数在 8193 到 16384 字节之间的语句数』
	第 436 页的『max_data_received_31999 － 接收的出站字节数在 16385 到 31999 字节之间的语句数』
	第 437 页的『max_data_received_64000 － 接收的出站字节数在 32000 到 64000 字节之间的语句数』
	第 438 页的『max_data_received_gt64000 － 接收的出站字节数高于 64000 的语句数』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
dcs_dbase（续）	第 428 页的『max_data_sent_128 － 发送的出站字节数在 1 到 128 字节之间的语句数』 第 429 页的『max_data_sent_256 － 发送的出站字节数在 129 到 256 字节之间的语句数』 第 430 页的『max_data_sent_512 － 发送的出站字节数在 257 到 512 字节之间的语句数』 第 431 页的『max_data_sent_1024 － 发送的出站字节数在 513 到 1024 字节之间的语句数』 第 432 页的『max_data_sent_2048 － 发送的出站字节数在 1025 到 2048 字节之间的语句数』 第 433 页的『max_data_sent_4096 － 发送的出站字节数在 2049 到 4096 字节之间的语句数』 第 433 页的『max_data_sent_8192 － 发送的出站字节数在 4097 到 8192 字节之间的语句数』 第 434 页的『max_data_sent_16384 － 发送的出站字节数在 8193 到 16384 字节之间的语句数』 第 435 页的『max_data_sent_31999 － 发送的出站字节数在 16385 到 31999 字节之间的语句数』 第 436 页的『max_data_sent_64000 － 发送的出站字节数在 32000 到 64000 字节之间的语句数』 第 437 页的『max_data_sent_gt64000 － 发送的出站字节数高于 64000 的语句数』 第 438 页的『max_network_time_1_ms － 网络时间最多为 1 ms 的语句数』 第 439 页的『max_network_time_4_ms － 网络时间在 1 到 4 ms 之间的语句数』 第 439 页的『max_network_time_16_ms － 网络时间在 4 到 16 ms 之间的语句数』 第 440 页的『max_network_time_100_ms － 网络时间在 16 到 100 ms 之间的语句数』 第 440 页的『max_network_time_500_ms － 网络时间在 100 到 500 ms 之间的语句数』 第 441 页的『max_network_time_gt500_ms － 网络时间大于 500 ms 的语句数』 第 442 页的『network_time_bottom － 语句的最短网络时间』 第 441 页的『network_time_top － 语句的最长网络时间』 第 425 页的『outbound_bytes_sent － 发送的出站字节数』 第 357 页的『rollback_sql_stmts － 尝试的回滚语句数』 第 337 页的『rows_selected － 选择的行数』 第 420 页的『sql_stmts － 尝试的 SQL 语句数』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
dcs_stmt	第 446 页的『blocking_cursor – 分块游标』 第 370 页的『creator – 应用程序创建者』 第 442 页的『elapsed_exec_time – 语句执行耗用时间』 第 374 页的『fetch_count – 成功的访存数』 第 419 页的『gw_exec_time – DB2 Connect 网关处理所耗用的时间』 第 443 页的『host_response_time – 主机响应时间』 第 424 页的『inbound_bytes_received – 接收的入站字节数』 第 426 页的『inbound_bytes_sent – 发送的入站字节数』 第 444 页的『num_transmissions_group – 传输次数组』 第 444 页的『num_transmissions – 传输次数』 第 425 页的『outbound_bytes_received – 接收的出站字节数』 第 425 页的『outbound_bytes_sent – 发送的出站字节数』 第 367 页的『package_name – 程序包名称』 第 375 页的『query_card_estimate – 行查询数估计』 第 376 页的『query_cost_estimate – 查询成本估计』 第 369 页的『section_number – 节号』 第 372 页的『stmt_elapsed_time – 最新语句耗用时间』 第 366 页的『stmt_operation/operation – 语句操作』 第 371 页的『stmt_start – 语句操作开始时间戳记』 第 371 页的『stmt_stop – 语句操作停止时间戳记』 第 373 页的『stmt_text – SQL 动态语句文本』
detail_log	第 285 页的『current_active_log – 当前活动日志文件编号』 第 285 页的『current_archive_log – 当前归档日志文件编号』 第 284 页的『first_active_log – 第一个活动日志文件编号』 第 284 页的『last_active_log – 最后一个活动日志文件编号』 第 175 页的『node_number – 节点号』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
dynsql	第 339 页的『int_rows_deleted － 删除的内部行数』
	第 341 页的『int_rows_inserted － 插入的内部行数』
	第 340 页的『int_rows_updated － 更新的内部行数』
	第 388 页的『num_compilations － 语句编译次数』
	第 388 页的『num_executions － 语句执行次数』
	第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』
	第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』
	第 227 页的『pool_index_l_reads － 缓冲池索引逻辑读取数』
	第 229 页的『pool_index_p_reads － 缓冲池索引物理读取数』
	第 224 页的『pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数』
	第 225 页的『pool_temp_data_p_reads － 缓冲池临时数据物理读取数』
	第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』
	第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』
	第 252 页的『pool_temp_xda_l_reads － 逻辑读取的缓冲池临时 XDA 数据页数』
	第 253 页的『pool_temp_xda_p_reads － 物理读取的缓冲池临时 XDA 数据页数』
	第 249 页的『pool_xda_l_reads － 逻辑读取的缓冲池 XDA 数据页数』
	第 250 页的『pool_xda_p_reads － 物理读取的缓冲池 XDA 数据页数』
	第 389 页的『prep_time_best － 语句最短编译时间』
	第 388 页的『prep_time_worst － 语句最长编译时间』
	第 338 页的『rows_read － 读取的行数』
	第 337 页的『rows_written － 写入的行数』
	第 203 页的『sort_overflows － 排序溢出数』
	第 373 页的『stmt_sorts － 语句排序数』
	第 373 页的『stmt_text － SQL 动态语句文本』
	第 389 页的『total_exec_time － 执行语句所耗用的时间』
	第 202 页的『total_sort_time － 总排序时间』
	第 396 页的『total_sys_cpu_time － 语句的总系统 CPU 』
	第 397 页的『total_usr_cpu_time － 语句的总用户 CPU 』
dynsql_list	第 149 页的『db_name － 数据库名称』
	第 150 页的『db_path － 数据库路径』
fcm	第 210 页的『buff_free － 当前可用的 FCM 缓冲区数』
	第 210 页的『buff_free_bottom － 最少可用 FCM 缓冲区数』
	第 211 页的『ch_free － 当前可用的通道数』
	第 211 页的『ch_free_bottom － 最低可用通道数』
fcm_node	第 211 页的『connection_status － 连接状态』
	第 175 页的『node_number － 节点号』
	第 212 页的『total_buffers_sent － 发送的总 FCM 缓冲区数』
	第 212 页的『total_buffers_rcvd － 接收到的总 FCM 缓冲区数』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
hadr	第 407 页的『hadr_connect_status － HADR 连接状态监视元素』 第 407 页的『hadr_connect_time － HADR 连接时间监视元素』 第 408 页的『hadr_heartbeat － HADR 脉动信号监视元素』 第 409 页的『hadr_local_host － HADR 本地主机监视元素』 第 409 页的『hadr_local_service － HADR 本地服务监视元素』 第 414 页的『hadr_log_gap － HADR 日志间隔』 第 411 页的『hadr_primary_log_file － HADR 主日志文件监视元素』 第 412 页的『hadr_primary_log_lsn － HADR 主日志 LSN 监视元素』 第 412 页的『hadr_primary_log_page － HADR 主日志页监视元素』 第 410 页的『hadr_remote_host － HADR 远程主机监视元素』 第 410 页的『hadr_remote_instance － HADR 远程实例监视元素』 第 410 页的『hadr_remote_service － HADR 远程服务监视元素』 第 405 页的『hadr_role － HADR 角色』 第 412 页的『hadr_standby_log_file － HADR 备用日志文件监视元素』 第 413 页的『hadr_standby_log_lsn － HADR 备用日志 LSN 监视元素』 第 413 页的『hadr_standby_log_page － HADR 备用日志页监视元素』 第 406 页的『hadr_state － HADR 状态监视元素』 第 406 页的『hadr_syncmode － HADR 同步方式监视元素』 第 411 页的『hadr_timeout － HADR 超时监视元素』
lock	第 181 页的『data_partition_id － 数据分区标识监视元素』 第 298 页的『lock_attributes － 锁定属性』 第 300 页的『lock_count － 锁定计数』 第 301 页的『lock_current_mode － 转换前的原始锁定方式』 第 295 页的『lock_escalation － 锁定升级』 第 300 页的『lock_hold_count － 锁定挂起计数』 第 290 页的『lock_mode － 锁定方式』 第 298 页的『lock_name － 锁定名称』 第 293 页的『lock_object_name － 锁定对象名称』 第 292 页的『lock_object_type － 等待的锁定对象类型』 第 299 页的『lock_release_flags － 锁定释放标志』 第 291 页的『lock_status － 锁定状态』 第 175 页的『node_number － 节点号』 第 341 页的『table_file_id － 表文件标识』 第 333 页的『table_name － 表名称』 第 334 页的『table_schema － 表模式名称』 第 312 页的『tablespace_name － 表空间名称』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
lock_wait	第 305 页的『agent_id_holding_lock － 挂起锁定的代理程序标识』 第 306 页的『appl_id_holding_lk － 挂起锁定的应用程序标识』 第 298 页的『lock_attributes － 锁定属性』 第 301 页的『lock_current_mode － 转换前的原始锁定方式』 第 295 页的『lock_escalation － 锁定升级』 第 290 页的『lock_mode － 锁定方式』 第 296 页的『lock_mode_requested － 请求的锁定方式』 第 298 页的『lock_name － 锁定名称』 第 292 页的『lock_object_type － 等待的锁定对象类型』 第 299 页的『lock_release_flags － 锁定释放标志』 第 304 页的『lock_wait_start_time － 锁定等待开始时间戳记』 第 175 页的『node_number － 节点号』 第 382 页的『ss_number － 子节号』 第 333 页的『table_name － 表名称』 第 334 页的『table_schema － 表模式名称』 第 312 页的『tablespace_name － 表空间名称』 第 181 页的『data_partition_id － 数据分区标识监视元素』
memory_pool	第 175 页的『node_number － 节点号』 第 196 页的『pool_cur_size － 内存池的当前大小』 第 195 页的『pool_id － 内存池标识』 第 196 页的『pool_secondary_id － 内存池辅助标记』 第 197 页的『pool_config_size － 内存池的已配置大小』 第 198 页的『pool_watermark － 内存池水位标记』
progress	第 218 页的『progress_completed_units － 完成的进度工作单元数』 第 216 页的『progress_description － 进度描述』 第 216 页的『progress_seq_num － 进度序号』 第 216 页的『progress_start_time － 进度开始时间』 第 217 页的『progress_total_units － 进度工作单元总数』 第 217 页的『progress_work_metric － 进度工作度量』
progress_list	第 215 页的『progress_list_cur_seq_num － 当前进度列表序号』 第 215 页的『progress_list_attr － 当前进度列表属性』
rollforward	第 175 页的『node_number － 节点号』 第 309 页的『rf_type － 前滚类型』 第 310 页的『rf_log_num － 正在前滚的日志』 第 310 页的『rf_status － 日志阶段』 第 309 页的『rf_timestamp － 前滚时间戳记』 第 309 页的『ts_name － 正在前滚的表空间』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
stmt	第 390 页的『agents_top － 创建的代理程序数』 第 446 页的『blocking_cursor － 分块游标』 第 368 页的『consistency_token － 程序包一致性标记』 第 370 页的『creator － 应用程序创建者』 第 369 页的『cursor_name － 游标名称』 第 390 页的『degree_parallelism － 并行度』 第 374 页的『fetch_count － 成功的访存数』 第 339 页的『int_rows_deleted － 删除的内部行数』 第 341 页的『int_rows_inserted － 插入的内部行数』 第 340 页的『int_rows_updated － 更新的内部行数』 第 390 页的『num_agents － 正在处理语句的代理程序数』 第 367 页的『package_name － 程序包名称』 第 368 页的『package_version_id － 程序包版本』 第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』 第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』 第 227 页的『pool_index_l_reads － 缓冲池索引逻辑读取数』 第 229 页的『pool_index_p_reads － 缓冲池索引物理读取数』 第 224 页的『pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数』 第 225 页的『pool_temp_data_p_reads － 缓冲池临时数据物理读取数』 第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』 第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』 第 252 页的『pool_temp_xda_l_reads － 逻辑读取的缓冲池临时 XDA 数据页数』 第 253 页的『pool_temp_xda_p_reads － 物理读取的缓冲池临时 XDA 数据页数』 第 249 页的『pool_xda_l_reads － 逻辑读取的缓冲池 XDA 数据页数』 第 250 页的『pool_xda_p_reads － 物理读取的缓冲池 XDA 数据页数』 第 375 页的『query_card_estimate － 行查询数估计』 第 376 页的『query_cost_estimate － 查询成本估计』 第 338 页的『rows_read － 读取的行数』 第 337 页的『rows_written － 写入的行数』 第 369 页的『section_number － 节号』 第 203 页的『sort_overflows － 排序溢出数』 第 372 页的『stmt_elapsed_time － 最新语句耗用时间』 第 363 页的『stmt_node_number － 语句节点』 第 366 页的『stmt_operation/operation － 语句操作』 第 373 页的『stmt_sorts － 语句排序数』 第 371 页的『stmt_start － 语句操作开始时间戳记』 第 371 页的『stmt_stop － 语句操作停止时间戳记』 第 393 页的『stmt_sys_cpu_time － 语句使用的系统 CPU 时间』 第 373 页的『stmt_text － SQL 动态语句文本』 第 365 页的『stmt_type － 语句类型』 第 392 页的『stmt_usr_cpu_time － 语句使用的用户 CPU 时间』 第 202 页的『total_sort_time － 总排序时间』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
stmt_transmissions	第 442 页的『elapsed_exec_time － 语句执行耗用时间』 第 443 页的『host_response_time － 主机响应时间』 第 428 页的『max_data_received_128 － 接收的出站字节数在 1 到 128 字节之间的语句数』 第 429 页的『max_data_received_256 － 接收的出站字节数在 129 到 256 字节之间的语句数』 第 430 页的『max_data_received_512 － 接收的出站字节数在 257 到 512 字节之间的语句数』 第 431 页的『max_data_received_1024 － 接收的出站字节数在 513 到 1024 字节之间的语句数』 第 432 页的『max_data_received_2048 － 发送的出站字节数在 1025 到 2048 字节之间的语句数』 第 433 页的『max_data_received_4096 － 接收的出站字节数在 2049 到 4096 字节之间的语句数』 第 434 页的『max_data_received_8192 － 接收的出站字节数在 4097 到 8192 字节之间的语句数』 第 435 页的『max_data_received_16384 － 发送的出站字节数在 8193 到 16384 字节之间的语句数』 第 436 页的『max_data_received_31999 － 接收的出站字节数在 16385 到 31999 字节之间的语句数』 第 437 页的『max_data_received_64000 － 接收的出站字节数在 32000 到 64000 字节之间的语句数』 第 438 页的『max_data_received_gt64000 － 接收的出站字节数高于 64000 的语句数』 第 428 页的『max_data_sent_128 － 发送的出站字节数在 1 到 128 字节之间的语句数』 第 429 页的『max_data_sent_256 － 发送的出站字节数在 129 到 256 字节之间的语句数』 第 430 页的『max_data_sent_512 － 发送的出站字节数在 257 到 512 字节之间的语句数』 第 431 页的『max_data_sent_1024 － 发送的出站字节数在 513 到 1024 字节之间的语句数』 第 432 页的『max_data_sent_2048 － 发送的出站字节数在 1025 到 2048 字节之间的语句数』 第 433 页的『max_data_sent_4096 － 发送的出站字节数在 2049 到 4096 字节之间的语句数』 第 433 页的『max_data_sent_8192 － 发送的出站字节数在 4097 到 8192 字节之间的语句数』 第 434 页的『max_data_sent_16384 － 发送的出站字节数在 8193 到 16384 字节之间的语句数』 第 435 页的『max_data_sent_31999 － 发送的出站字节数在 16385 到 31999 字节之间的语句数』 第 436 页的『max_data_sent_64000 － 发送的出站字节数在 32000 到 64000 字节之间的语句数』 第 437 页的『max_data_sent_gt64000 － 发送的出站字节数高于 64000 的语句数』
stmt_transmissions（续）	第 438 页的『max_network_time_1_ms － 网络时间最多为 1 ms 的语句数』 第 439 页的『max_network_time_4_ms － 网络时间在 1 到 4 ms 之间的语句数』 第 439 页的『max_network_time_16_ms － 网络时间在 4 到 16 ms 之间的语句数』 第 440 页的『max_network_time_100_ms － 网络时间在 16 到 100 ms 之间的语句数』 第 440 页的『max_network_time_500_ms － 网络时间在 100 到 500 ms 之间的语句数』 第 441 页的『max_network_time_gt500_ms － 网络时间大于 500 ms 的语句数』 第 441 页的『network_time_top － 语句的最长网络时间』 第 442 页的『network_time_bottom － 语句的最短网络时间』 第 425 页的『outbound_bytes_received － 接收的出站字节数』 第 425 页的『outbound_bytes_sent － 发送的出站字节数』 第 426 页的『outbound_bytes_sent_top － 发送的最大出站字节数』 第 427 页的『outbound_bytes_received_top － 接收的最大出站字节数』 第 427 页的『outbound_bytes_sent_bottom － 发送的最小出站字节数』 第 428 页的『outbound_bytes_received_bottom － 接收的最小出站字节数』 第 421 页的『sql_chains － 尝试的 SQL 链数』 第 420 页的『sql_stmts － 尝试的 SQL 语句数』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
subsection	第 338 页的『rows_read － 读取的行数』
	第 337 页的『rows_written － 写入的行数』
	第 384 页的『ss_exec_time － 子节执行耗用时间』
	第 383 页的『ss_node_number － 子节节点号』
	第 382 页的『ss_number － 子节号』
	第 383 页的『ss_status － 子节状态』
	第 396 页的『ss_sys_cpu_time － 子节使用的系统 CPU 时间』
	第 395 页的『ss_usr_cpu_time － 子节使用的用户 CPU 时间』
	第 385 页的『tq_cur_send_spills － 当前溢出的表队列缓冲区数』
	第 387 页的『tq_id_waiting_on － 在表队列的节点上等待』
	第 387 页的『tq_max_send_spills － 最大表队列缓冲区溢出数』
	第 384 页的『tq_node_waited_for － 在表队列上等待节点』
	第 386 页的『tq_rows_read － 从表队列读取的行数』
	第 386 页的『tq_rows_written － 写至表队列的行数』
	第 385 页的『tq_tot_send_spills － 溢出的表队列缓冲区总数』
	第 384 页的『tq_wait_for_any － 在表队列上等待发送任何节点』
table	第 343 页的『data_object_pages － 数据对象页数』
	第 181 页的『data_partition_id － 数据分区标识监视元素』
	第 343 页的『index_object_pages － 索引对象页数』
	第 344 页的『lob_object_pages － LOB 对象页数』
	第 344 页的『long_object_pages － 长整型对象页数』
	第 339 页的『overflow_accesses － 对溢出记录的访问次数』
	第 342 页的『page_reorgs － 页重组数』
	第 338 页的『rows_read － 读取的行数』
	第 337 页的『rows_written － 写入的行数』
	第 341 页的『table_file_id － 表文件标识』
	第 333 页的『table_name － 表名称』
	第 334 页的『table_schema － 表模式名称』
	第 311 页的『tablespace_id － 表空间标识』
	第 181 页的『data_partition_id － 数据分区标识监视元素』
	第 332 页的『table_type － 表类型』
	第 253 页的『xda_object_pages － XDA 对象页数』
table_list	第 150 页的『db_conn_time － 数据库激活时间戳记』
	第 149 页的『db_name － 数据库名称』
	第 150 页的『db_path － 数据库路径』
	第 398 页的『input_db_alias － 输入数据库别名』
	第 397 页的『last_reset － 最后复位时间戳记』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
table_reorg	第 181 页的『data_partition_id － 数据分区标识监视元素』 第 347 页的『reorg_completion － 重组完成标志』 第 347 页的『reorg_current_counter － 重组进度』 第 348 页的『reorg_end － 表重组结束时间』 第 348 页的『reorg_index_id － 用于重组表的索引』 第 347 页的『reorg_max_counter － 重组总量』 第 347 页的『reorg_max_phase － 最大重组阶段』 第 346 页的『reorg_phase － 重组阶段』 第 346 页的『reorg_phase_start － 重组阶段开始时间』 第 348 页的『reorg_start － 表重组开始时间』 第 346 页的『reorg_status － 表重组状态』 第 349 页的『reorg_tbspc_id － 用来重组表或数据分区的表空间』 第 345 页的『reorg_type － 表重组属性』 第 349 页的『reorg_rows_compressed － 压缩行数』 第 349 页的『reorg_rows_rejected_for_compression － 拒绝压缩行数』
tablespace	第 257 页的『direct_read_reqs － 直接读取请求数』 第 258 页的『direct_read_time － 直接读取时间』 第 255 页的『direct_reads － 直接从数据库进行读取的读取操作数』 第 257 页的『direct_write_reqs － 直接写入请求数』 第 259 页的『direct_write_time － 直接写入时间』 第 256 页的『direct_writes － 直接写入数据库的写入操作数』 第 233 页的『files_closed － 关闭的数据库文件数』 第 331 页的『fs_caching － 文件系统高速缓存』 第 239 页的『pool_async_data_read_reqs － 缓冲池异步读取请求数』 第 234 页的『pool_async_data_reads － 缓冲池异步数据读取数』 第 235 页的『pool_async_data_writes － 缓冲池异步数据写入数』 第 239 页的『pool_async_index_read_reqs － 缓冲池异步索引读取请求数』 第 236 页的『pool_async_index_reads － 缓冲池异步索引读取数』 第 235 页的『pool_async_index_writes － 缓冲池异步索引写入数』 第 237 页的『pool_async_read_time － 缓冲池异步读取时间』 第 238 页的『pool_async_write_time － 缓冲池异步写入时间』 第 247 页的『pool_async_xda_read_reqs － 缓冲池异步 XDA 读取请求数』 第 247 页的『pool_async_xda_reads － 缓冲池异步 XDA 数据读取数』 第 248 页的『pool_async_xda_writes － 缓冲池异步 XDA 数据写入数』 第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』 第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』 第 226 页的『pool_data_writes － 缓冲池数据页写入数』 第 227 页的『pool_index_l_reads － 缓冲池索引逻辑读取数』 第 229 页的『pool_index_p_reads － 缓冲池索引物理读取数』 第 231 页的『pool_index_writes － 缓冲池索引写入数』 第 241 页的『pool_no_victim_buffer － 缓冲池没有牺牲缓冲区的次数』 第 232 页的『pool_read_time － 缓冲池物理读总时间』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
tablespace（续）	第 224 页的『pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数』
	第 225 页的『pool_temp_data_p_reads － 缓冲池临时数据物理读取数』
	第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』
	第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』
	第 252 页的『pool_temp_xda_l_reads － 逻辑读取的缓冲池临时 XDA 数据页数』
	第 253 页的『pool_temp_xda_p_reads － 物理读取的缓冲池临时 XDA 数据页数』
	第 233 页的『pool_write_time － 缓冲池物理写入总时间』
	第 249 页的『pool_xda_l_reads － 逻辑读取的缓冲池 XDA 数据页数』
	第 250 页的『pool_xda_p_reads － 物理读取的缓冲池 XDA 数据页数』
	第 251 页的『pool_xda_writes － 缓冲池 XDA 数据写入数』
	第 318 页的『tablespace_auto_resize_enabled － 能够自动调整大小』
	第 313 页的『tablespace_content_type － 表空间内容类型』
	第 315 页的『tablespace_cur_pool_id － 当前使用的缓冲池』
	第 314 页的『tablespace_extent_size － 表空间的扩展数据块大小』
	第 311 页的『tablespace_id － 表空间标识』
	第 312 页的『tablespace_name － 表空间名称』
	第 315 页的『tablespace_next_pool_id － 将在下次启动时使用的缓冲池』
	第 314 页的『tablespace_page_size － 表空间的页大小』
	第 315 页的『tablespace_prefetch_size － 表空间的预取大小』
	第 320 页的『tablespace_rebalancer_mode － 重新平衡程序方式』
	第 313 页的『tablespace_type － 表空间类型』
	第 317 页的『tablespace_using_auto_storage － 使用自动存储器』
tablespace_container	第 328 页的『container_accessible － 容器的可访问性』
	第 326 页的『container_id － 容器标识』
	第 326 页的『container_name － 容器名称』
	第 328 页的『container_stripe_set － 条带集』
	第 327 页的『container_total_pages － 容器中的总页数』
	第 326 页的『container_type － 容器类型』
	第 327 页的『container_usable_pages － 容器中的可用页数』
tablespace_list	第 150 页的『db_conn_time － 数据库激活时间戳记』
	第 149 页的『db_name － 数据库名称』
	第 150 页的『db_path － 数据库路径』
	第 398 页的『input_db_alias － 输入数据库别名』
	第 397 页的『last_reset － 最后复位时间戳记』

表 14. 快照监视器逻辑数据组和监视元素（续）

快照逻辑数据组	监视元素
tablespace_nodeinfo	第 318 页的『tablespace_current_size － 当前表空间大小』 第 317 页的『tablespace_free_pages － 表空间中的空闲页数』 第 319 页的『tablespace_increase_size － 增加字节大小』 第 319 页的『tablespace_increase_size_percent － 增加大小（以百分比计）』 第 318 页的『tablespace_initial_size － 初始表空间大小』 第 320 页的『tablespace_last_resize_failed － 上一次调整大小尝试失败』 第 320 页的『tablespace_last_resize_time － 上次成功调整大小的时间』 第 319 页的『tablespace_max_size － 最大表空间大小』 第 325 页的『tablespace_min_recovery_time － 前滚的最短恢复时间』 第 325 页的『tablespace_num_containers － 表空间中的容器数目』 第 323 页的『tablespace_num_quiescers － 停顿者数目』 第 328 页的『tablespace_num_ranges － 表空间映射中的范围数』 第 317 页的『tablespace_page_top － 表空间高水位标记』 第 317 页的『tablespace_pending_free_pages － 表空间中的暂挂可用页数』 第 315 页的『tablespace_prefetch_size － 表空间的预取大小』 第 322 页的『tablespace_rebalancer_extents_processed － 重新平衡程序已经处理的扩展数据块数』 第 321 页的『tablespace_rebalancer_extents_remaining － 重新平衡程序要处理的扩展数据块总数』 第 322 页的『tablespace_rebalancer_last_extent_moved － 重新平衡程序移动的最后一个扩展数据块』 第 322 页的『tablespace_rebalancer_priority － 当前重新平衡程序优先级』 第 321 页的『tablespace_rebalancer_restart_time － 重新平衡程序重新启动时间』 第 321 页的『tablespace_rebalancer_start_time － 重新平衡程序启动时间』 第 313 页的『tablespace_state － 表空间状态』 第 325 页的『tablespace_state_change_object_id － 状态更改对象标识』 第 325 页的『tablespace_state_change_ts_id － 状态更改表空间标识』 第 316 页的『tablespace_total_pages － 表空间中的总页数』 第 316 页的『tablespace_usable_pages － 表空间中的可用页数』 第 316 页的『tablespace_used_pages － 表空间中的已使用页数』
tablespace_quiescer	第 323 页的『quiescer_agent_id － 停顿者代理程序标识』 第 323 页的『quiescer_auth_id － 停顿者用户授权标识』 第 324 页的『quiescer_obj_id － 停顿者对象标识』 第 324 页的『quiescer_state － 停顿者状态』 第 324 页的『quiescer_ts_id － 停顿者表空间标识』
tablespace_range	第 330 页的『range_adjustment － 范围调整』 第 331 页的『range_container_id － 范围容器』 第 330 页的『range_end_stripe － 结束条带』 第 330 页的『range_max_extent － 范围中的最大扩展数据块』 第 329 页的『range_max_page_number － 范围中的最大页』 第 331 页的『range_num_containers － 范围中的容器数』 第 329 页的『range_number － 范围编号』 第 331 页的『range_offset － 范围位移』 第 330 页的『range_start_stripe － 起始条带』 第 329 页的『range_stripe_set_number － 条带集编号』

表 14. 快照监视器逻辑数据组和监视元素 (续)

快照逻辑数据组	监视元素
utility_info	第 213 页的『utility_dbname － 实用程序操作的数据库』 第 213 页的『utility_id － 实用程序标识』 第 219 页的『utility_invoker_type － 实用程序调用者类型』 第 218 页的『utility_state － 实用程序状态』 第 214 页的『utility_type － 实用程序类型』 第 214 页的『utility_priority － 实用程序优先级』 第 214 页的『utility_start_time － 实用程序启动时间』 第 215 页的『utility_description － 实用程序描述』 第 175 页的『node_number － 节点号』

事件类型至逻辑数据组的映射

事件监视器输出由一个有序的逻辑数据分组序列组成。不管事件监视器类型如何，输出记录总是包含相同的起始逻辑数据组。它们为逻辑数据组设置了框架，这些逻辑数据组的存在于否取决于事件监视器记录的事件类型。

对于文件和管道事件监视器，可能会对任何连接生成事件记录，并且事件记录可能会因此以混合顺序出现在流中。这意味着您可能获得连接 1 的事务事件，之后紧跟连接 2 的连接事件。但是，属于单个连接或单个事件的记录将以逻辑顺序出现。例如，语句记录（语句结尾）总是在事务记录（UOW 结尾）之前，如果有的话。同样，死锁事件总是在死锁中涉及的每个连接的死锁连接事件记录之前。**应用程序标识**或**应用程序句柄（agent_id）**可用来将记录与连接相匹配。

通常会对与数据库的每个连接写入连接头事件。对于带有详细信息的死锁事件监视器，仅当发生死锁时才写入它们。在此情况下，将仅对死锁参与者（而不是与数据库的所有连接）写入连接头事件。

逻辑数据分组将按以下四个不同级别排序：监视器、序言、内容和结尾。下面详细描述每个级别，包括对应的事件类型和逻辑数据组。

监视器:

将对所有事件监视器生成监视器级别的信息。它包含事件监视器元数据。

表 15. 事件监视器数据流：监视器部分

事件类型	逻辑数据组	可用信息
监视器级别	event_log_stream_header	标识事件监视器的版本级别和字节顺序。应用程序可使用此头来确定是否能够处理 evmon 输出流。

序言:

在激活事件监视器时将生成序言信息。

表 16. 事件监视器数据流：序言部分

事件类型	逻辑数据组	可用信息
日志头	event_log_header	跟踪的特征，如服务器类型和内存布局。
数据库头	event_db_header	数据库名称、路径和激活时间。
事件监视器启动	event_start	启动或重新启动监视器的时间。
连接头	event_connheader	每个当前事件对应一个连接头，包括连接时间和应用程序名称。仅对连接、语句、事务和死锁事件监视器生成事件连接头。带有详细信息的死锁事件监视器仅在发生死锁时生成连接头。

内容:

特定于事件监视器的指定事件类型的信息出现在内容部分。

表 17. 事件监视器数据流：内容部分

事件类型	逻辑数据组	可用信息
语句事件	event_stmt	语句级别数据，包括动态语句的文本。语句事件监视器不记录访存。
子节事件	event_subsection	子节级别数据。
事务事件	event_xact	事务级别数据。
连接事件	event_conn	连接级别数据。
死锁事件	event_deadlock	死锁级别数据。
死锁的连接事件	event_dlconn	死锁涉及的每个连接对应一个死锁的连接事件，包括涉及的应用程序和处于争用状态的锁定。
带有详细信息的死锁的连接事件	event_detailed_dlconn, lock	死锁涉及的每个连接对应一个带有详细信息的死锁的连接事件，包括涉及的应用程序、处于争用状态的锁定、当前语句信息和应用程序争用挂起的其他锁定。
溢出	event_overflow	丢失的记录数 － 在写程序不能与（非分块）事件监视器保持一致时生成。
带有详细信息的死锁的历史纪录	event_stmt_history	列示死锁涉及的任何工作单元中执行的语句列表。
带有详细信息的死锁的历史记录值	event_data_value	event_stmt_history 列表中的语句的参数标记。

结尾:

结尾信息将在数据库释放时（最后一个应用程序断开连接后）生成:

表 18. 事件监视器数据流: 结尾部分

事件类型	逻辑数据组	可用信息
数据库事件	event_db	数据库管理器级别数据。
缓冲池事件	event_bufferpool	缓冲池级别数据。
表空间事件	event_tablespace	表空间级别数据。
表事件	event_table	表级别数据。

相关概念:

- 第 3 页的『数据库系统监视器数据结构』
- 第 55 页的『事件监视器』

相关任务:

- 第 57 页的『收集关于数据库系统事件的信息』

相关参考:

- 第 75 页的『事件监视器样本输出』

事件监视器逻辑数据组和监视元素

下表列示事件监视可能返回的逻辑数据分组和监视元素。

表 19. 事件监视器逻辑数据组和监视元素

事件逻辑数据组	监视元素名称
event_bufferpool	第 243 页的『bp_name – 缓冲池名称』 第 222 页的『bp_id – 缓冲池标识』 第 149 页的『db_name – 数据库名称』 第 150 页的『db_path – 数据库路径』 第 257 页的『direct_read_reqs – 直接读取请求数』 第 258 页的『direct_read_time – 直接读取时间』 第 255 页的『direct_reads – 直接从数据库进行读取的读取操作数』 第 257 页的『direct_write_reqs – 直接写入请求数』 第 259 页的『direct_write_time – 直接写入时间』 第 256 页的『direct_writes – 直接写入数据库的写入操作数』 第 402 页的『event_time – 事件时间』 第 403 页的『evmon_activates – 事件监视器激活数』 第 402 页的『evmon_flushes – 事件监视器清空数』 第 233 页的『files_closed – 关闭的数据库文件数』 第 402 页的『partial_record – 部分记录』 第 239 页的『pool_async_data_read_reqs – 缓冲池异步读取请求数』 第 234 页的『pool_async_data_reads – 缓冲池异步数据读取数』 第 235 页的『pool_async_data_writes – 缓冲池异步数据写入数』 第 236 页的『pool_async_index_reads – 缓冲池异步索引读取数』 第 235 页的『pool_async_index_writes – 缓冲池异步索引写入数』 第 237 页的『pool_async_read_time – 缓冲池异步读取时间』 第 238 页的『pool_async_write_time – 缓冲池异步写入时间』 第 222 页的『pool_data_l_reads – 缓冲池数据页逻辑读取数』 第 224 页的『pool_data_p_reads – 缓冲池数据页物理读取数』 第 226 页的『pool_data_writes – 缓冲池数据页写入数』 第 227 页的『pool_index_l_reads – 缓冲池索引逻辑读取数』 第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』 第 231 页的『pool_index_writes – 缓冲池索引写入数』 第 232 页的『pool_read_time – 缓冲池物理读总时间』 第 233 页的『pool_write_time – 缓冲池物理写入总时间』

表 19. 事件监视器逻辑数据组和监视元素（续）

事件逻辑数据组	监视元素名称
event_conn	第 352 页的『acc_curs_blk － 接受的块游标请求数』 第 158 页的『agent_id － 应用程序句柄（代理程序标识）』 第 163 页的『appl_id － 应用程序标识』 第 173 页的『appl_priority － 应用程序代理程序优先级』 第 174 页的『appl_priority_type － 应用程序优先级类型』 第 267 页的『appl_section_inserts － 段插入数监视元素』 第 266 页的『appl_section_lookups － 段查询』 第 174 页的『authority_lvl － 用户权限级别』 第 363 页的『binds_precompiles － 尝试的绑定次数／预编译次数』 第 261 页的『cat_cache_inserts － 目录高速缓存插入数』 第 260 页的『cat_cache_lookups － 目录高速缓存查询数』 第 261 页的『cat_cache_overflows － 目录高速缓存溢出数』 第 356 页的『commit_sql_stmts － 尝试的落实语句数』 第 359 页的『ddl_sql_stmts － 数据定义语言（DDL）SQL 语句数』 第 288 页的『deadlocks － 检测到的死锁数』 第 257 页的『direct_read_reqs － 直接读取请求数』 第 258 页的『direct_read_time － 直接读取时间』 第 255 页的『direct_reads － 直接从数据库进行读取的读取操作数』 第 257 页的『direct_write_reqs － 直接写入请求数』 第 259 页的『direct_write_time － 直接写入时间』 第 256 页的『direct_writes － 直接写入数据库的写入操作数』 第 151 页的『disconn_time － 数据库释放时间戳记』 第 354 页的『dynamic_sql_stmts － 尝试的动态 SQL 语句数』 第 355 页的『failed_sql_stmts － 失败的语句操作数』 第 208 页的『hash_join_overflows － 散列连接溢出数』 第 209 页的『hash_join_small_overflows － 散列连接小溢出数』 第 359 页的『int_auto_rebinds － 内部自动重新绑定次数』 第 360 页的『int_commits － 内部落实数』 第 362 页的『int_deadlock_rollbacks － 死锁导致的内部回滚数』 第 361 页的『int_rollbacks － 内部回滚数』 第 339 页的『int_rows_deleted － 删除的内部行数』 第 341 页的『int_rows_inserted － 插入的内部行数』 第 340 页的『int_rows_updated － 更新的内部行数』 第 295 页的『lock_escalation － 锁定升级』 第 294 页的『lock_timeouts － 锁定超时次数』 第 303 页的『lock_wait_time － 等待锁定的时间』 第 302 页的『lock_waits － 锁定等待数』 第 402 页的『partial_record － 部分记录』 第 265 页的『pkg_cache_inserts － 程序包高速缓存插入数』 第 263 页的『pkg_cache_lookups － 程序包高速缓存查询数』 第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』 第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』 第 226 页的『pool_data_writes － 缓冲池数据页写入数』

表 19. 事件监视器逻辑数据组和监视元素（续）

事件逻辑数据组	监视元素名称
event_conn（续）	第 227 页的『pool_index_l_reads － 缓冲池索引逻辑读取数』 第 229 页的『pool_index_p_reads － 缓冲池索引物理读取数』 第 231 页的『pool_index_writes － 缓冲池索引写入数』 第 232 页的『pool_read_time － 缓冲池物理读总时间』 第 224 页的『pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数』 第 225 页的『pool_temp_data_p_reads － 缓冲池临时数据物理读取数』 第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』 第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』 第 233 页的『pool_write_time － 缓冲池物理写入总时间』 第 244 页的『prefetch_wait_time － 等待预取的时间』 第 271 页的『priv_workspace_num_overflows － 专用工作空间溢出数』 第 272 页的『priv_workspace_section_inserts － 专用工作空间段插入数』 第 272 页的『priv_workspace_section_lookups － 专用工作空间段查询数』 第 271 页的『priv_workspace_size_top － 最大专用工作空间大小』 第 351 页的『rej_curs_blk － 拒绝的块游标请求数』 第 357 页的『rollback_sql_stmts － 尝试的回滚语句数』 第 338 页的『rows_read － 读取的行数』 第 337 页的『rows_selected － 选择的行数』 第 340 页的『int_rows_updated － 更新的内部行数』 第 337 页的『rows_written － 写入的行数』 第 357 页的『select_sql_stmts － 执行的 SELECT SQL 语句数』 第 166 页的『sequence_no － 序号』 第 269 页的『shr_workspace_num_overflows － 共享工作空间溢出数』 第 270 页的『shr_workspace_section_inserts － 共享工作空间段插入数』 第 269 页的『shr_workspace_section_lookups － 共享工作空间段查询数』 第 268 页的『shr_workspace_size_top － 最大共享工作空间大小』 第 203 页的『sort_overflows － 排序溢出数』 第 354 页的『static_sql_stmts － 尝试的静态 SQL 语句数』 第 394 页的『system_cpu_time － 系统 CPU 时间』 第 206 页的『total_hash_joins － 散列连接总数』 第 208 页的『total_hash_loops － 总散列循环数』 第 193 页的『total_sec_cons － 辅助连接数』 第 202 页的『total_sort_time － 总排序时间』 第 201 页的『total_sorts － 总排序数』 第 358 页的『uid_sql_stmts － 执行的 Update/Insert/Delete SQL 语句数』 第 244 页的『unread_prefetch_pages － 未读取的预取页数』 第 394 页的『user_cpu_time － 用户 CPU 时间』 第 290 页的『x_lock_escals － 互斥锁定升级数』 第 364 页的『xquery_stmts － 尝试的 XQuery 语句数』

表 19. 事件监视器逻辑数据组和监视元素（续）

事件逻辑数据组	监视元素名称
event_connheader	第 158 页的『agent_id － 应用程序句柄（代理程序标识）』
	第 163 页的『appl_id － 应用程序标识』
	第 163 页的『appl_name － 应用程序名称』
	第 166 页的『auth_id － 授权标识』
	第 168 页的『client_db_alias － 应用程序使用的数据库别名』
	第 167 页的『client_nname － 客户机的配置 NNAME 』
	第 171 页的『client_pid － 客户机进程标识』
	第 172 页的『client_platform － 客户机操作平台』
	第 168 页的『client_prdid － 客户机产品／版本标识』
	第 172 页的『client_protocol － 客户机通信协议』
	第 161 页的『codepage_id － 应用程序使用的代码页的标识』
	第 151 页的『conn_time － 数据库连接时间』
	第 171 页的『corr_token － DRDA 关联标记』
	第 170 页的『execution_id － 用户登录标识』
	第 175 页的『node_number － 节点号』
	第 166 页的『sequence_no － 序号』
	第 173 页的『territory_code － 数据库地域代码』
event_connmemuse	第 175 页的『node_number － 节点号』
	第 196 页的『pool_cur_size － 内存池的当前大小』
	第 195 页的『pool_id － 内存池标识』
	第 196 页的『pool_secondary_id － 内存池辅助标记』
	第 197 页的『pool_config_size － 内存池的已配置大小』
	第 198 页的『pool_watermark － 内存池水位标记』
event_data_value	第 403 页的『evmon_activates － 事件监视器激活数』
	第 296 页的『deadlock_id － 死锁事件标识』
	第 297 页的『deadlock_node － 发生死锁的分区号』
	第 297 页的『participant_no － 死锁参与者』
	第 380 页的『stmt_value_type － 值类型』
	第 376 页的『stmt_history_id － 语句历史记录标识』
	第 380 页的『stmt_value_isnull － 包含空值』
	第 381 页的『stmt_value_data － 值数据』
	第 381 页的『stmt_value_index － 值索引』
	第 382 页的『stmt_value_isreoptvalue － 用于语句重新优化的变量』

表 19. 事件监视器逻辑数据组和监视元素（续）

事件逻辑数据组	监视元素名称
event_db	第 206 页的『active_hash_joins – 活动散列连接数』
	第 267 页的『appl_section_inserts – 段插入数监视元素』
	第 266 页的『appl_section_lookups – 段查询』
	第 363 页的『binds_precompiles – 尝试的绑定次数／预编译次数』
	第 261 页的『cat_cache_inserts – 目录高速缓存插入数』
	第 260 页的『cat_cache_lookups – 目录高速缓存查询数』
	第 261 页的『cat_cache_overflows – 目录高速缓存溢出数』
	第 262 页的『cat_cache_size_top – 目录高速缓存高水位标记』
	第 153 页的『catalog_node – 目录节点号』
	第 152 页的『catalog_node_name – 目录节点网络名』
	第 356 页的『commit_sql_stmts – 尝试的落实语句数』
	第 176 页的『connections_top – 最大并行连接数』
	第 273 页的『db_heap_top – 分配的最大数据库堆』
	第 359 页的『ddl_sql_stmts – 数据定义语言（DDL）SQL 语句数』
	第 288 页的『deadlocks – 检测到的死锁数』
	第 257 页的『direct_read_reqs – 直接读取请求数』
	第 258 页的『direct_read_time – 直接读取时间』
	第 255 页的『direct_reads – 直接从数据库进行读取的读取操作数』
	第 257 页的『direct_write_reqs – 直接写入请求数』
	第 259 页的『direct_write_time – 直接写入时间』
	第 256 页的『direct_writes – 直接写入数据库的写入操作数』
	第 151 页的『disconn_time – 数据库释放时间戳记』
	第 354 页的『dynamic_sql_stmts – 尝试的动态 SQL 语句数』
	第 403 页的『evmon_activates – 事件监视器激活数』
	第 402 页的『evmon_flushes – 事件监视器清空数』
	第 355 页的『failed_sql_stmts – 失败的语句操作数』
	第 233 页的『files_closed – 关闭的数据库文件数』
	第 208 页的『hash_join_overflows – 散列连接溢出数』
	第 209 页的『hash_join_small_overflows – 散列连接小溢出数』
	第 359 页的『int_auto_rebinds – 内部自动重新绑定次数』
	第 360 页的『int_commits – 内部落实数』
	第 361 页的『int_rollbacks – 内部回滚数』
	第 339 页的『int_rows_deleted – 删除的内部行数』
	第 341 页的『int_rows_inserted – 插入的内部行数』
	第 340 页的『int_rows_updated – 更新的内部行数』
	第 289 页的『lock_escals – 锁定升级数』
	第 294 页的『lock_timeouts – 锁定超时次数』
	第 303 页的『lock_wait_time – 等待锁定的时间』
	第 302 页的『lock_waits – 锁定等待数』
	第 279 页的『log_held_by_dirty_pages – 脏页占用的日志空间量』
	第 281 页的『log_read_time – 日志读取时间』
	第 276 页的『log_reads – 读取的日志页数』
	第 280 页的『log_to_redo_for_recovery – 要为恢复重做的日志量』
	第 280 页的『log_write_time – 日志写入时间』
	第 277 页的『log_writes – 写入的日志页数』
	第 282 页的『num_log_read_io – 日志读取数』
	第 281 页的『num_log_write_io – 日志写入次数』

表 19. 事件监视器逻辑数据组和监视元素（续）

事件逻辑数据组	监视元素名称
event_db（续）	第 402 页的『partial_record － 部分记录』 第 265 页的『pkg_cache_inserts － 程序包高速缓存插入数』 第 263 页的『pkg_cache_lookups － 程序包高速缓存查询数』 第 265 页的『pkg_cache_num_overflows － 程序包高速缓存溢出数』 第 266 页的『pkg_cache_size_top － 程序包高速缓存高水位标记』 第 239 页的『pool_async_data_read_reqs － 缓冲池异步读取请求数』 第 234 页的『pool_async_data_reads － 缓冲池异步数据读取数』 第 235 页的『pool_async_data_writes － 缓冲池异步数据写入数』 第 239 页的『pool_async_index_read_reqs － 缓冲池异步索引读取请求数』 第 236 页的『pool_async_index_reads － 缓冲池异步索引读取数』 第 235 页的『pool_async_index_writes － 缓冲池异步索引写入数』 第 237 页的『pool_async_read_time － 缓冲池异步读取时间』 第 238 页的『pool_async_write_time － 缓冲池异步写入时间』 第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』 第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』 第 226 页的『pool_data_writes － 缓冲池数据页写入数』 第 240 页的『pool_drty_pg_steal_clns － 触发的缓冲池牺牲页清除程序数』 第 242 页的『pool_drty_pg_thrsh_clns － 触发的缓冲池阈值清除程序数』 第 227 页的『pool_index_l_reads － 缓冲池索引逻辑读取数』 第 229 页的『pool_index_p_reads － 缓冲池索引物理读取数』 第 231 页的『pool_index_writes － 缓冲池索引写入数』 第 240 页的『pool_lsn_gap_clns － 触发的缓冲池日志空间清除程序数』 第 241 页的『pool_no_victim_buffer － 缓冲池没有牺牲缓冲区的次数』 第 232 页的『pool_read_time － 缓冲池物理读总时间』 第 224 页的『pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数』 第 225 页的『pool_temp_data_p_reads － 缓冲池临时数据物理读取数』 第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』 第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』 第 233 页的『pool_write_time － 缓冲池物理写入总时间』 第 207 页的『post_shrthreshold_hash_joins － 阈值后散列连接数』 第 205 页的『post_shrthreshold_sorts － 阈值后排序数』 第 244 页的『prefetch_wait_time － 等待预取的时间』 第 271 页的『priv_workspace_num_overflows － 专用工作空间溢出数』 第 272 页的『priv_workspace_section_inserts － 专用工作空间段插入数』 第 272 页的『priv_workspace_section_lookups － 专用工作空间段查询数』 第 271 页的『priv_workspace_size_top － 最大专用工作空间大小』 第 357 页的『rollback_sql_stmts － 尝试的回滚语句数』 第 335 页的『rows_deleted － 删除的行数』 第 335 页的『rows_inserted － 插入的行数』 第 338 页的『rows_read － 读取的行数』 第 337 页的『rows_selected － 选择的行数』 第 336 页的『rows_updated － 更新的行数』

表 19. 事件监视器逻辑数据组和监视元素（续）

事件逻辑数据组	监视元素名称
event_db（续）	第 274 页的『sec_log_used_top － 使用的最大辅助日志空间』 第 357 页的『select_sql_stmts － 执行的 SELECT SQL 语句数』 第 147 页的『server_platform － 服务器操作系统』 第 269 页的『shr_workspace_num_overflows － 共享工作空间溢出数』 第 270 页的『shr_workspace_section_inserts － 共享工作空间段插入数』 第 269 页的『shr_workspace_section_lookups － 共享工作空间段查询数』 第 268 页的『shr_workspace_size_top － 最大共享工作空间大小』 第 203 页的『sort_overflows － 排序溢出数』 第 354 页的『static_sql_stmts － 尝试的静态 SQL 语句数』 第 275 页的『tot_log_used_top － 使用的最大总日志空间』 第 186 页的『total_cons － 数据库激活以后的连接数』 第 206 页的『total_hash_joins － 散列连接总数』 第 208 页的『total_hash_loops － 总散列循环数』 第 202 页的『total_sort_time － 总排序时间』 第 201 页的『total_sorts － 总排序数』 第 358 页的『uid_sql_stmts － 执行的 Update/Insert/Delete SQL 语句数』 第 244 页的『unread_prefetch_pages － 未读取的预取页数』 第 290 页的『x_lock_escals － 互斥锁定升级数』 第 364 页的『xquery_stmts － 尝试的 XQuery 语句数』
event_dbheader	第 151 页的『conn_time － 数据库连接时间』 第 149 页的『db_name － 数据库名称』 第 150 页的『db_path － 数据库路径』
event_dbmemuse	第 175 页的『node_number － 节点号』 第 196 页的『pool_cur_size － 内存池的当前大小』 第 195 页的『pool_id － 内存池标识』 第 197 页的『pool_config_size － 内存池的已配置大小』 第 198 页的『pool_watermark － 内存池水位标记』
event_deadlock	第 296 页的『deadlock_id － 死锁事件标识』 第 297 页的『deadlock_node － 发生死锁的分区号』 第 295 页的『dl_conns － 死锁中涉及的连接数』 第 403 页的『evmon_activates － 事件监视器激活数』 第 307 页的『rolled_back_agent_id － 回滚的代理程序』 第 307 页的『rolled_back_appl_id － 回滚的应用程序』 第 298 页的『rolled_back_participant_no － 回滚的应用程序参与者』 第 307 页的『rolled_back_sequence_no － 回滚的序号』 第 372 页的『start_time － 事件启动时间』

表 19. 事件监视器逻辑数据组和监视元素 (续)

事件逻辑数据组	监视元素名称
event_detailed_dlconn	第 158 页的『agent_id － 应用程序句柄（代理程序标识）』
	第 163 页的『appl_id － 应用程序标识』
	第 306 页的『appl_id_holding_lk － 挂起锁定的应用程序标识』
	第 446 页的『blocking_cursor － 分块游标』
	第 368 页的『consistency_token － 程序包一致性标记』
	第 370 页的『creator － 应用程序创建者』
	第 369 页的『cursor_name － 游标名称』
	第 296 页的『deadlock_id － 死锁事件标识』
	第 297 页的『deadlock_node － 发生死锁的分区号』
	第 181 页的『data_partition_id － 数据分区标识监视元素』
	第 403 页的『evmon_activates － 事件监视器激活数』
	第 295 页的『lock_escalation － 锁定升级』
	第 290 页的『lock_mode － 锁定方式』
	第 296 页的『lock_mode_requested － 请求的锁定方式』
	第 293 页的『lock_node － 锁定节点』
	第 293 页的『lock_object_name － 锁定对象名称』
	第 292 页的『lock_object_type － 等待的锁定对象类型』
	第 304 页的『lock_wait_start_time － 锁定等待开始时间戳记』
	第 287 页的『locks_held － 挂起的锁定数』
	第 298 页的『locks_in_list － 报告的锁定数』
	第 367 页的『package_name － 程序包名称』
	第 368 页的『package_version_id － 程序包版本』
	第 297 页的『participant_no － 死锁参与者』
	第 297 页的『participant_no_holding_lk － 挂起对应用程序所需对象的锁定的参与者』
	第 369 页的『section_number － 节号』
	第 166 页的『sequence_no － 序号』
	第 306 页的『sequence_no_holding_lk － 挂起锁定的序号』
	第 372 页的『start_time － 事件启动时间』
	第 366 页的『stmt_operation/operation － 语句操作』
	第 373 页的『stmt_text － SQL 动态语句文本』
	第 365 页的『stmt_type － 语句类型』
	第 333 页的『table_name － 表名称』
	第 334 页的『table_schema － 表模式名称』
	第 312 页的『tablespace_name － 表空间名称』

表 19. 事件监视器逻辑数据组和监视元素（续）

事件逻辑数据组	监视元素名称
event_dlconn	第 158 页的『agent_id － 应用程序句柄（代理程序标识）』 第 163 页的『appl_id － 应用程序标识』 第 306 页的『appl_id_holding_lk － 挂起锁定的应用程序标识』 第 296 页的『deadlock_id － 死锁事件标识』 第 181 页的『data_partition_id － 数据分区标识监视元素』 第 297 页的『deadlock_node － 发生死锁的分区号』 第 403 页的『evmon_activates － 事件监视器激活数』 第 298 页的『lock_attributes － 锁定属性』 第 300 页的『lock_count － 锁定计数』 第 301 页的『lock_current_mode － 转换前的原始锁定方式』 第 295 页的『lock_escalation － 锁定升级』 第 300 页的『lock_hold_count － 锁定挂起计数』 第 290 页的『lock_mode － 锁定方式』 第 296 页的『lock_mode_requested － 请求的锁定方式』 第 298 页的『lock_name － 锁定名称』 第 293 页的『lock_node － 锁定节点』 第 293 页的『lock_object_name － 锁定对象名称』 第 292 页的『lock_object_type － 等待的锁定对象类型』 第 299 页的『lock_release_flags － 锁定释放标志』 第 304 页的『lock_wait_start_time － 锁定等待开始时间戳记』 第 297 页的『participant_no － 死锁参与者』 第 297 页的『participant_no_holding_lk － 挂起对应用程序所需对象的锁定的参与者』 第 166 页的『sequence_no － 序号』 第 306 页的『sequence_no_holding_lk － 挂起锁定的序号』 第 372 页的『start_time － 事件启动时间』 第 333 页的『table_name － 表名称』 第 334 页的『table_schema － 表模式名称』 第 312 页的『tablespace_name － 表空间名称』
event_log_header	第 400 页的『byte_order － 事件数据的字节顺序』 第 161 页的『codepage_id － 应用程序使用的代码页的标识』 第 401 页的『event_monitor_name － 事件监视器名称』 第 399 页的『num_nodes_in_db2_instance － 分区中的节点数』 第 146 页的『server_prdid － 服务器产品／版本标识』 第 145 页的『server_instance_name － 服务器实例名称』 第 173 页的『territory_code － 数据库地域代码』 第 401 页的『version － 监视器数据版本』
event_overflow	第 400 页的『count － 事件监视器溢出数』 第 400 页的『first_overflow_time － 第一次事件溢出时间』 第 400 页的『last_overflow_time － 最后一次事件溢出时间』 第 175 页的『node_number － 节点号』
event_start	第 372 页的『start_time － 事件启动时间』

表 19. 事件监视器逻辑数据组和监视元素 (续)

事件逻辑数据组	监视元素名称
event_stmt	第 158 页的『agent_id － 应用程序句柄（代理程序标识）』
	第 390 页的『agents_top － 创建的代理程序数』
	第 163 页的『appl_id － 应用程序标识』
	第 446 页的『blocking_cursor － 分块游标』
	第 368 页的『consistency_token － 程序包一致性标记』
	第 370 页的『creator － 应用程序创建者』
	第 369 页的『cursor_name － 游标名称』
	第 374 页的『fetch_count － 成功的访存数』
	第 339 页的『int_rows_deleted － 删除的内部行数』
	第 341 页的『int_rows_inserted － 插入的内部行数』
	第 340 页的『int_rows_updated － 更新的内部行数』
	第 367 页的『package_name － 程序包名称』
	第 368 页的『package_version_id － 程序包版本』
	第 402 页的『partial_record － 部分记录』
	第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』
	第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』
	第 227 页的『pool_index_l_reads － 缓冲池索引逻辑读取数』
	第 229 页的『pool_index_p_reads － 缓冲池索引物理读取数』
	第 224 页的『pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数』
	第 225 页的『pool_temp_data_p_reads － 缓冲池临时数据物理读取数』
	第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』
	第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』
	第 338 页的『rows_read － 读取的行数』
	第 337 页的『rows_written － 写入的行数』
	第 369 页的『section_number － 节号』
	第 166 页的『sequence_no － 序号』
	第 203 页的『sort_overflows － 排序溢出数』
	第 403 页的『sql_req_id － SQL 语句的请求标识』
	第 375 页的『sqlca － SQL 通信区（SQLCA）』
	第 372 页的『start_time － 事件启动时间』
	第 366 页的『stmt_operation/operation － 语句操作』
	第 373 页的『stmt_text － SQL 动态语句文本』
	第 365 页的『stmt_type － 语句类型』
	第 371 页的『stop_time － 事件停止时间』
	第 394 页的『system_cpu_time － 系统 CPU 时间』
	第 202 页的『total_sort_time － 总排序时间』
	第 201 页的『total_sorts － 总排序数』
	第 394 页的『user_cpu_time － 用户 CPU 时间』

表 19. 事件监视器逻辑数据组和监视元素（续）

事件逻辑数据组	监视元素名称
event_stmt_history	第 403 页的『evmon_activates － 事件监视器激活数』 第 296 页的『deadlock_id － 死锁事件标识』 第 297 页的『deadlock_node － 发生死锁的分区号』 第 297 页的『participant_no － 死锁参与者』 第 376 页的『stmt_history_id － 语句历史记录标识』 第 377 页的『stmt_first_use_time － 语句第一次使用时间』 第 377 页的『stmt_last_use_time － 语句上一次使用时间监视元素』 第 377 页的『stmt_lock_timeout － 语句锁定超时』 第 378 页的『stmt_isolation － 语句隔离』 第 367 页的『package_name － 程序包名称』 第 368 页的『package_version_id － 程序包版本』 第 369 页的『section_number － 节号』 第 370 页的『creator － 应用程序创建者』 第 365 页的『stmt_type － 语句类型』 第 373 页的『stmt_text － SQL 动态语句文本』 第 380 页的『comp_env_desc － 编译环境句柄』 第 378 页的『stmt_nest_level － 语句嵌套级别』 第 378 页的『stmt_invocation_id － 语句调用标识』 第 379 页的『stmt_query_id － 语句查询标识』 第 379 页的『stmt_source_id － 语句源标识』 第 166 页的『sequence_no － 序号』 第 380 页的『stmt_pkgcache_id － 语句程序包高速缓存标识』
event_subsection	第 158 页的『agent_id － 应用程序句柄（代理程序标识）』 第 390 页的『num_agents － 正在处理语句的代理程序数』 第 402 页的『partial_record － 部分记录』 第 384 页的『ss_exec_time － 子节执行耗用时间』 第 383 页的『ss_node_number － 子节节点号』 第 382 页的『ss_number － 子节号』 第 396 页的『ss_sys_cpu_time － 子节使用的系统 CPU 时间』 第 395 页的『ss_usr_cpu_time － 子节使用的用户 CPU 时间』 第 387 页的『tq_max_send_spills － 最大表队列缓冲区溢出数』 第 386 页的『tq_rows_read － 从表队列读取的行数』 第 386 页的『tq_rows_written － 写至表队列的行数』 第 385 页的『tq_tot_send_spills － 溢出的表队列缓冲区总数』

表 19. 事件监视器逻辑数据组和监视元素 (续)

事件逻辑数据组	监视元素名称
event_table	第 343 页的『data_object_pages – 数据对象页数』 第 402 页的『event_time – 事件时间』 第 403 页的『evmon_activates – 事件监视器激活数』 第 402 页的『evmon_flushes – 事件监视器清空数』 第 343 页的『index_object_pages – 索引对象页数』 第 344 页的『lob_object_pages – LOB 对象页数』 第 344 页的『long_object_pages – 长整型对象页数』 第 339 页的『overflow_accesses – 对溢出记录的访问次数』 第 342 页的『page_reorgs – 页重组数』 第 402 页的『partial_record – 部分记录』 第 338 页的『rows_read – 读取的行数』 第 337 页的『rows_written – 写入的行数』 第 333 页的『table_name – 表名称』 第 334 页的『table_schema – 表模式名称』 第 332 页的『table_type – 表类型』 第 181 页的『data_partition_id – 数据分区标识监视元素』

表 19. 事件监视器逻辑数据组和监视元素 (续)

事件逻辑数据组	监视元素名称
event_tablespace	第 257 页的『direct_read_reqs – 直接读取请求数』 第 258 页的『direct_read_time – 直接读取时间』 第 255 页的『direct_reads – 直接从数据库进行读取的读取操作数』 第 257 页的『direct_write_reqs – 直接写入请求数』 第 259 页的『direct_write_time – 直接写入时间』 第 256 页的『direct_writes – 直接写入数据库的写入操作数』 第 402 页的『event_time – 事件时间』 第 403 页的『evmon_activates – 事件监视器激活数』 第 402 页的『evmon_flushes – 事件监视器清空数』 第 233 页的『files_closed – 关闭的数据库文件数』 第 402 页的『partial_record – 部分记录』 第 239 页的『pool_async_data_read_reqs – 缓冲池异步读取请求数』 第 234 页的『pool_async_data_reads – 缓冲池异步数据读取数』 第 235 页的『pool_async_data_writes – 缓冲池异步数据写入数』 第 239 页的『pool_async_index_read_reqs – 缓冲池异步索引读取请求数』 第 236 页的『pool_async_index_reads – 缓冲池异步索引读取数』 第 235 页的『pool_async_index_writes – 缓冲池异步索引写入数』 第 237 页的『pool_async_read_time – 缓冲池异步读取时间』 第 238 页的『pool_async_write_time – 缓冲池异步写入时间』 第 222 页的『pool_data_l_reads – 缓冲池数据页逻辑读取数』 第 224 页的『pool_data_p_reads – 缓冲池数据页物理读取数』 第 226 页的『pool_data_writes – 缓冲池数据页写入数』 第 227 页的『pool_index_l_reads – 缓冲池索引逻辑读取数』 第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』 第 231 页的『pool_index_writes – 缓冲池索引写入数』 第 241 页的『pool_no_victim_buffer – 缓冲池没有牺牲缓冲区的次数』 第 232 页的『pool_read_time – 缓冲池物理读总时间』 第 224 页的『pool_temp_data_l_reads – 缓冲池临时数据逻辑读取数』 第 225 页的『pool_temp_data_p_reads – 缓冲池临时数据物理读取数』 第 228 页的『pool_temp_index_l_reads – 缓冲池临时索引逻辑读取数』 第 230 页的『pool_temp_index_p_reads – 缓冲池临时索引物理读取数』 第 233 页的『pool_write_time – 缓冲池物理写入总时间』 第 312 页的『tablespace_name – 表空间名称』

表 19. 事件监视器逻辑数据组和监视元素 (续)

事件逻辑数据组	监视元素名称
event_xact	第 158 页的『agent_id － 应用程序句柄（代理程序标识）』 第 163 页的『appl_id － 应用程序标识』 第 289 页的『lock_escals － 锁定升级数』 第 303 页的『lock_wait_time － 等待锁定的时间』 第 294 页的『locks_held_top － 挂起的最大锁定数』 第 402 页的『partial_record － 部分记录』 第 177 页的『prev_uow_stop_time － 上一个工作单元完成时间戳记』 第 338 页的『rows_read － 读取的行数』 第 337 页的『rows_written － 写入的行数』 第 166 页的『sequence_no － 序号』 第 394 页的『system_cpu_time － 系统 CPU 时间』 第 277 页的『uow_log_space_used － 使用的工作单元日志空间』 第 178 页的『uow_start_time － 工作单元开始时间戳记』 第 180 页的『uow_status － 工作单元状态』 第 179 页的『uow_stop_time － 工作单元停止时间戳记』 第 394 页的『user_cpu_time － 用户 CPU 时间』 第 290 页的『x_lock_escals － 互斥锁定升级数』
lock	第 298 页的『lock_attributes － 锁定属性』 第 300 页的『lock_count － 锁定计数』 第 301 页的『lock_current_mode － 转换前的原始锁定方式』 第 295 页的『lock_escalation － 锁定升级』 第 300 页的『lock_hold_count － 锁定挂起计数』 第 290 页的『lock_mode － 锁定方式』 第 298 页的『lock_name － 锁定名称』 第 293 页的『lock_object_name － 锁定对象名称』 第 292 页的『lock_object_type － 等待的锁定对象类型』 第 299 页的『lock_release_flags － 锁定释放标志』 第 291 页的『lock_status － 锁定状态』 第 175 页的『node_number － 节点号』 第 341 页的『table_file_id － 表文件标识』 第 333 页的『table_name － 表名称』 第 334 页的『table_schema － 表模式名称』 第 312 页的『tablespace_name － 表空间名称』 第 181 页的『data_partition_id － 数据分区标识监视元素』

表 19. 事件监视器逻辑数据组和监视元素 (续)

事件逻辑数据组	监视元素名称
sqlca	sqlcabc sqlcode sqlerrml sqlcaid sqlerrmc sqlerrp sqlerrd sqlwarn sqlstate

第 6 章 监视元素

数据库系统监视元素

系统监视器返回的监视元素分为下列几个类别:

- **标识**，用于要监视的数据库管理器、应用程序或数据库连接。
- 这些数据主要用于帮助您**配置**系统。
- 各个级别的数据库**活动**，包括数据库、应用程序、表或语句。此信息可用于活动监视、问题确定和性能分析。此信息还可用于配置。
- 有关 **DB2 Connect™** 应用程序的信息。包括在网关上运行的 DCS 应用程序、要执行的 SQL 语句和数据库连接。
- 有关**联合数据库系统**的信息。这包括有关由在 DB2 联合系统中运行的应用程序对数据源的总体访问的信息，以及由在联合服务器实例中运行的给定应用程序对数据源的访问的信息。

监视元素是以如下标准格式描述的:

元素标识
元素的名称。如果直接对数据流进行语法分析，则元素标识将是大写的，并且加上 SQLM_ELM_ 前缀。

元素类型
监视元素返回的信息类型。例如，db2start_time 监视元素返回时间戳记。

快照监视信息
如果监视元素返回快照监视信息，将显示带有下列字段的表。

- *快照级别*: 快照监视器可捕获的信息的级别。例如，appl_status 监视元素返回应用程序级别和锁定级别的信息。
- *逻辑数据分组*: 返回捕获的快照信息的逻辑数据组。如果直接对数据流进行语法分析，则逻辑数据组标识将是大写的，并且加上 SQLM_ELM_ 前缀。例如，appl_status 监视元素返回有关 appl_id_info 分组和 appl_lock_list 分组的信息。
- *监视开关*: 为获取此信息而必须设置的系统监视开关。如果开关为“基本”，则总是收集有关该监视元素的数据。

事件监视信息
如果监视元素是由事件监视器收集的，将显示带有下列字段的表。

- *事件类型*: 事件监视器可收集的信息的级别。必须使用此事件类型来创建事件监视器以收集此信息。例如，将为 CONNECTIONS 事件监视器收集 appl_status 监视元素。
- *逻辑数据分组*: 在其中返回捕获的事件信息的逻辑数据组。如果直接对数据流进行语法分析，则逻辑数据组标识将是大写的，并且加上 SQLM_ELM_ 前缀。例如，appl_status 监视元素返回有关 event_conn 分组的信息。

- *监视开关*： 为获取此信息而必须设置的系统监视开关。对于事件监视器，TIMESTAMP 开关是唯一能够限制事件数据收集的监视开关。如果此字段显示为虚线，则总是收集有关该监视元素的数据。

描述 对监视元素所收集数据的描述。

用法 有关在监视数据库系统时能够如何使用监视元素收集的信息的信息。

服务器标识和状态

服务器标识和状态监视元素

下列元素提供有关服务器的标识和状态信息：

- db2start_time － 启动数据库管理器时间戳记监视元素
- server_nname － 监视（服务器）数据库分区上的配置 NNAME 监视元素
- server_instance_name － 服务器实例名称监视元素
- server_db2_type － 受监视的（服务器）节点处的数据库管理器类型监视元素
- server_prdid － 服务器产品／版本标识监视元素
- server_version － 服务器版本监视元素
- service_level － 服务级别监视元素
- server_platform － 服务器操作系统监视元素
- product_name － 产品名称监视元素
- db2_status － DB2 实例的状态监视元素
- time_zone_disp － 时区偏移监视元素

db2start_time － 启动数据库管理器时间戳记

元素标识 db2start_time

元素类型 时间戳记

表 20. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 使用 `db2start` 命令启动数据库管理器的日期和时间。

用法 此元素可与 *time_stamp* 监视元素配合使用，以计算自数据库管理器启动直到获取快照所耗用的时间。

相关参考:

- 第 398 页的『time_stamp － 快照时间』

server_nname － 监视（服务器）数据库分区上的配置 NNAME

元素标识 server_nname

元素类型 信息

表 21. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

描述 将由数据库系统监视器监视的数据库分区的名称。

用法 此元素可用来标识要监视的数据库服务器分区。如果要将监视器输出保存在文件或数据库中以便以后进行分析，并且需要区分来自不同数据库服务器分区的数据，则此信息会非常有用。此数据库分区名称将根据 *nname* 配置参数确定。

此元素仅适用于具有 NetBIOS LAN 环境的 Windows 环境。

相关参考:

• 第 167 页的『client_nname － 客户机的配置 NNAME 』

server_instance_name － 服务器实例名称

元素标识 server_instance_name

元素类型 信息

表 22. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

表 23. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

描述 对其获取快照的数据库管理器实例的名称。

用法 如果同一系统上存在多个数据库管理器实例，则此数据项将用于唯一标识对其发出快照调用的实例。如果要将监视器输出保存在文件或数据库中以便以后进行分析，并且需要区分来自不同数据库管理器实例的数据，则将此信息与 *server_nname* 配合使用会非常有用。

相关参考:

• 第 144 页的『server_nname － 监视（服务器）数据库分区上的配置 NNAME 』

server_db2_type － 受监视的（服务器）节点上的数据库管理器类型

元素标识 server_db2_type

元素类型 信息

表 24. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

描述 标识要监视的数据库管理器的类型。

用法 它包含数据库管理器的下列配置类型的其中一个:

API 符号常量	命令行处理器输出
sqlf_nt_server	带本地客户机和远程客户机的数据库服务器
sqlf_nt_stand_req	带本地客户机的数据库服务器

API 符号常量是在包含文件 *sqlutil.h* 中定义的。

相关参考:

- 第 144 页的『server_nname － 监视（服务器）数据库分区上的配置 NNAME 』

server_prdid － 服务器产品／版本标识

元素标识 server_prdid

元素类型 信息

表 25. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

表 26. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

描述 正在服务器上运行的产品和版本。

用法 其格式为 PPPVVRRM，其中:

PPP 为 SQL
VV 标识两位版本号（版本只有一位时高位为 0）
RR 标识两位发行版号（发行版只有一位时高位为 0）
M 标识一位修改级别

相关参考:

- 第 168 页的『client_prdid － 客户机产品／版本标识』

server_version － 服务器版本

元素标识 server_version

元素类型 信息

表 27. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

描述 返回该信息的服务器的版本。

用法 此字段标识收集数据库系统监视器信息的数据库服务器的级别。这允许应用程序根据返回数据的服务器级别来解释数据。有效值为:

SQLM_DBMON_VERSION1 数据由 DB2 版本 1 返回

SQLM_DBMON_VERSION2 数据由 DB2 版本 2 返回

SQLM_DBMON_VERSION5 数据由 DB2 通用数据库版本 5 返回

SQLM_DBMON_VERSION5_2
数据由 DB2 通用数据库版本 5.2 返回

SQLM_DBMON_VERSION6 数据由 DB2 通用数据库版本 6 返回

SQLM_DBMON_VERSION7 数据由 DB2 通用数据库版本 7 返回

SQLM_DBMON_VERSION8 数据由 DB2 通用数据库版本 8 返回

相关参考:

- 第 146 页的『server_prdid － 服务器产品／版本标识』

service_level － 服务级别

元素标识 service_level

元素类型 信息

表 28. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 这是 DB2 实例的当前校正服务级别。

server_platform － 服务器操作系统

元素标识 server_platform

元素类型 信息

表 29. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 30. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 运行数据库服务器的操作系统。

用法 此元素可用于远程应用程序的问题确定。可在头文件 *sqlmon.h* 中找到此字段的值。

相关参考:

- 第 172 页的『client_platform － 客户机操作平台』
- 第 153 页的『db_location － 数据库位置』

product_name － 产品名称

元素标识 product_name

元素类型　　信息

表 31. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述　正在运行的 DB2 实例的版本的详细信息。

db2_status － DB2 实例的状态

元素标识　　db2_status

元素类型　　信息

表 32. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述　数据库管理器的实例的当前状态。

用法　可使用此元素来确定数据库管理器实例的状态。

此元素的值包括:

API 常量	值	描述
SQLM_DB2_ACTIVE	0	数据库管理器实例是活动的。
SQLM_DB2_QUIESCE_PEND	1	实例和实例中的数据库处于停顿－暂挂状态。不允许新建与任何实例数据库的连接，也不能启动新的工作单元。根据停顿请求，将允许活动工作单元完成或立即回滚。
SQLM_DB2_QUIESCED	2	实例和实例中的数据库已停顿。不允许新建与任何实例数据库的连接，也不能启动新的工作单元。

相关参考:

- 第 152 页的『db_status － 数据库状态』

time_zone_disp － 时区偏移

元素标识　　time_zone_disp

元素类型　　信息

表 33. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

描述　表示本地时区与格林威治标准时间（GMT）之间的偏移的秒数。

用法　数据库系统监视器报告的所有时间为 GMT，而此偏移计算本地时间。

数据库标识和状态

数据库标识和状态监视元素

下列元素提供有关数据库的标识和状态信息:

- db_name – 数据库名称监视元素
- db_path – 数据库路径监视元素
- db_conn_time – 数据库激活时间戳记监视元素
- conn_time – 数据库连接的时间监视元素
- disconn_time – 数据库取消激活时间戳记监视元素
- db_status – 数据库的状态监视元素
- catalog_node_name – 目录节点网络名监视元素
- db_location – 数据库位置监视元素
- catalog_node – 目录节点号监视元素
- last_backup – 最近的备份时间戳记监视元素
- num_db_storage_paths – 自动存储器路径数监视元素num_db_storage_paths – 自动存储器路径数监视元素
- db_storage_path – 自动存储器路径监视元素db_storage_path – 自动存储器路径监视元素

db_name – 数据库名称

元素标识 db_name

元素类型 信息

表 34. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl_id_info	基本
应用程序	appl_remote	基本
表空间	tablespace_list	缓冲池
缓冲池	bufferpool	缓冲池
表	table_list	表
锁	db_lock_list	基本
动态 SQL	dynsql_list	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

表 35. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbheader	-

描述 对其收集信息或应用程序连接至的数据库的真实名称。这是创建时给定的数据库名称。

用法 可使用此元素来标识应用数据的特定数据库。

对于未使用 DB2 Connect 连接至主机或 AS/400® 和 iSeries™ 数据库服务器的应用程序，可将此元素与 *db_path* 监视元素配合使用来唯一标识该数据库，并协助使监视器提供的信息的不同级别相关。

相关参考:

- 第 397 页的『last_reset – 最后复位时间戳记』
- 第 398 页的『input_db_alias – 输入数据库别名』
- 第 168 页的『client_db_alias – 应用程序使用的数据库别名』
- 第 150 页的『db_path – 数据库路径』

db_path – 数据库路径

元素标识 db_path

元素类型 信息

表 36. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl_id_info	基本
表空间	tablespace_list	缓冲池
缓冲池	bufferpool	缓冲池
表	table_list	表
锁	db_lock_list	基本
动态 SQL	dynsql_list	基本

表 37. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbheader	-

描述 数据库在被监视系统上的存储位置的完整路径。

用法 此元素可与 *db_name* 监视元素配合使用以标识应用数据的特定数据库。

相关参考:

- 第 398 页的『input_db_alias – 输入数据库别名』
- 第 149 页的『db_name – 数据库名称』

db_conn_time – 数据库激活时间戳记

元素标识 db_conn_time

元素类型 时间戳记

表 38. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	时间戳记
表空间	tablespace_list	缓冲池，时间戳记
表	table_list	时间戳记

描述 在数据库级别第一次连接至数据库的日期和时间，或者发出激活数据库命令的日期和时间。

用法 将此元素与 disconn_time 监视元素配合使用以计算总连接时间。

相关参考:

- 第 176 页的『appl_con_time － 连接请求启动时间戳记』
- 第 398 页的『time_stamp － 快照时间』
- 第 151 页的『conn_time － 数据库连接时间』

conn_time － 数据库连接时间

元素标识 conn_time

元素类型 时间戳记

表 39. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbheader	-
连接	event_connheader	-

描述 在数据库级别第一次连接至数据库的日期和时间，或者发出激活数据库命令的日期和时间。

用法 将此元素与 disconn_time 监视元素配合使用来计算自出现以下情况后所耗用的时间:
- 数据库处于活动状态（用于数据库级别的信息）
- 连接处于活动状态（用于连接级别的信息）

相关参考:

- 第 150 页的『db_conn_time － 数据库激活时间戳记』
- 第 151 页的『disconn_time － 数据库释放时间戳记』

disconn_time － 数据库释放时间戳记

元素标识 disconn_time

元素类型 时间戳记

表 40. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 应用程序与数据库在数据库级别断开连接的日期和时间，这是应用程序最后一次断开连接。

用法 使用此元素来计算自出现以下情况以来耗用的时间:
- 数据库处于活动状态（用于数据库级别的信息）
- 连接处于活动状态（用于连接级别的信息）

db_status – 数据库状态

元素标识 db_status

元素类型 信息

表 41. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 数据库的当前状态。

用法 可使用此元素来确定数据库的状态。

此字段的值是:

API 常量	值	描述
SQLM_DB_ACTIVE	0	数据库是活动的。
SQLM_DB_QUIESCE_PEND	1	数据库处于停顿 – 暂挂状态。不允许新建与数据库的连接，也不能启动新的工作单元。根据停顿请求，将允许活动工作单元完成或立即回滚。
SQLM_DB_QUIESCED	2	数据库已停顿。不允许新建与数据库的连接，也不能启动新的工作单元。
SQLM_DB_ROLLFWD	3	数据库正在进行前滚。

相关参考:

- 第 148 页的『db2_status – DB2 实例的状态』

catalog_node_name – 目录节点网络名

元素标识 catalog_node_name

元素类型 信息

表 42. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 43. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 目录节点的网络名。

用法 使用此元素来确定数据库的位置。

db_location － 数据库位置

元素标识 db_location

元素类型 信息

表 44. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 与应用程序相关的数据库的位置。

用法 确定有关获取快照的应用程序的数据库服务器的相对位置。值包括:
- SQLM_LOCAL
- SQLM_REMOTE

相关参考:

- 第 147 页的『server_platform － 服务器操作系统』

catalog_node － 目录节点号

元素标识 catalog_node

元素类型 信息

表 45. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 46. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 存储数据库目录表的节点的编号。

用法 目录节点是存储所有系统目录表的节点。对系统目录表的所有访问都必须通过此节点进行。

相关参考:

- 『BACKUP DATABASE command』（*Command Reference*）

last_backup － 上次备份时间戳记

元素标识 last_backup

元素类型 时间戳记

表 47. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	时间戳记

描述 完成上次数据库备份的日期和时间。

用法 可使用此元素来帮助您标识最近没有备份的数据库，或者标识最新的数据库备份文件。如果数据库从未进行备份，则此时间戳记将初始化为零。

num_db_storage_paths － 自动存储器路径数

元素标识 num_db_storage_paths

元素类型 信息

表 48. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 此元素显示与此数据库相关联的自动存储器路径的数目。

用法 可将此元素与 db_storage_path 监视元素配合使用来标识与此数据库相关联的存储器路径。

db_storage_path － 自动存储器路径

元素标识 db_storage_path

元素类型 信息

表 49. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	基本

描述 此元素显示数据库用于放置自动存储器表空间的位置的完整路径。一个数据库可以有 0 个或多个相关联的存储器路径。

用法 可将此元素与 num_db_storage_paths 监视元素配合使用来标识与此数据库相关联的存储器路径。

相关参考:

- 第 154 页的『sto_path_free_sz － 自动存储器路径可用空间』
- 第 156 页的『fs_id － 唯一文件系统标识号』
- 第 155 页的『fs_total_size － 文件系统总大小』
- 第 157 页的『fs_type － 文件系统类型』
- 第 155 页的『fs_used_size － 文件系统上的已用空间量』

sto_path_free_sz － 自动存储器路径可用空间

元素标识 fs_free_size

元素类型 信息

表 50. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

描述 此元素显示存储器路径指向的文件系统上的可用空间量。如果多个存储器路径指向同一个文件系统，则不会在它们之间划分可用大小。

用法 可以将此元素与下列元素配合使用以收集每个节点的数据库空间利用率数据:

- db_storage_path
- fs_used_size
- fs_total_size
- fs_id
- fs_type

相关参考:

- 第 154 页的『db_storage_path – 自动存储器路径』
- 第 156 页的『fs_id – 唯一文件系统标识号』
- 第 155 页的『fs_used_size – 文件系统上的已用空间量』
- 第 155 页的『fs_total_size – 文件系统总大小』
- 第 157 页的『fs_type – 文件系统类型』

fs_used_size – 文件系统上的已用空间量

元素标识 fs_used_size

元素类型 信息

表 51. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

描述 此元素显示存储路径指向的文件系统上的已用空间量。

用法 可以将此元素与下列元素配合使用以收集数据库空间利用率数据:

- db_storage_path
- sto_path_free_sz
- fs_total_size
- fs_id
- fs_type

相关参考:

- 第 154 页的『db_storage_path – 自动存储器路径』
- 第 154 页的『sto_path_free_sz – 自动存储器路径可用空间』
- 第 156 页的『fs_id – 唯一文件系统标识号』
- 第 155 页的『fs_total_size – 文件系统总大小』
- 第 157 页的『fs_type – 文件系统类型』
- 第 154 页的『num_db_storage_paths – 自动存储器路径数』

fs_total_size – 文件系统总大小

元素标识 fs_total_size

元素类型　　信息

表 52. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

描述　此元素显示存储路径指向的文件系统的容量。

用法　可以将此元素与下列元素配合使用以收集数据库空间利用率数据:

- db_storage_path
- sto_path_free_sz
- fs_used_size
- fs_id
- fs_type

相关参考:

- 第 154 页的『db_storage_path － 自动存储器路径』
- 第 154 页的『sto_path_free_sz － 自动存储器路径可用空间』
- 第 156 页的『fs_id － 唯一文件系统标识号』
- 第 157 页的『fs_type － 文件系统类型』
- 第 155 页的『fs_used_size － 文件系统上的已用空间量』

fs_id － 唯一文件系统标识号

元素标识　　fs_id

元素类型　　信息

表 53. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

描述　此元素显示操作系统为存储路径指向的文件系统提供的唯一标识号。

用法　可以将此元素与下列元素配合使用以收集数据库空间利用率数据:

- db_storage_path
- sto_path_free_sz
- fs_used_size
- fs_total_size
- fs_type

相关参考:

- 第 154 页的『db_storage_path － 自动存储器路径』
- 第 154 页的『sto_path_free_sz － 自动存储器路径可用空间』
- 第 155 页的『fs_total_size － 文件系统总大小』
- 第 157 页的『fs_type － 文件系统类型』
- 第 155 页的『fs_used_size － 文件系统上的已用空间量』

fs_type － 文件系统类型

元素标识 fs_type

元素类型 信息

表 54. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

描述 此元素显示存储路径指向的文件系统的类型。此文件系统类型是由操作系统提供的。

用法 可以将此元素与下列元素配合使用以收集数据库空间利用率数据:

- db_storage_path
- sto_path_free_sz
- fs_used_size
- fs_total_size
- fs_id

相关参考:

- 第 154 页的『db_storage_path － 自动存储器路径』
- 第 154 页的『sto_path_free_sz － 自动存储器路径可用空间』
- 第 156 页的『fs_id － 唯一文件系统标识号』
- 第 155 页的『fs_total_size － 文件系统总大小』
- 第 155 页的『fs_used_size － 文件系统上的已用空间量』

应用程序标识和状态

应用程序标识和状态监视元素

下列元素提供有关数据库及其相关应用程序的信息。

- agent_id － 应用程序句柄（代理程序标识）监视元素
- appl_status － 应用程序状态监视元素
- codepage_id － 应用程序使用的代码页标识监视元素
- status_change_time － 应用程序状态更改时间监视元素
- appl_id_oldest_xact － 带有最旧事务的应用程序监视元素
- smallest_log_avail_node － 带有最少可用日志空间的节点监视元素
- appl_name － 应用程序名称监视元素
- appl_id － 应用程序标识监视元素
- sequence_no － 序号监视元素
- auth_id － 授权标识监视元素
- session_auth_id － 会话授权标识监视元素session_auth_id － 会话授权标识监视元素
- client_nname － 客户机的配置 NNAME 监视元素

- client_prdid – 客户机产品 / 版本标识监视元素
- client_db_alias – 应用程序使用的数据库别名监视元素
- host_prdid – 主机产品 / 版本标识监视元素
- outbound_appl_id – 出站应用程序标识监视元素
- outbound_sequence_no – 出站序号监视元素
- execution_id – 用户登录标识监视元素
- corr_token – DRDA 相关性标记监视元素
- client_pid – 客户机进程标识监视元素
- client_platform – 客户机操作平台监视元素
- client_protocol – 客户机通信协议监视元素
- territory_code – 数据库地域代码监视元素
- appl_priority – 应用程序代理程序优先级监视元素
- appl_priority_type – 应用程序优先级类型监视元素
- authority_lvl – 用户权限级别监视元素
- node_number – 节点号监视元素
- data_partition_id – 数据分区标识监视元素data_partition_id – 数据分区标识监视元素
- coord_node – 协调节点监视元素
- appl_con_time – 连接请求开始时间戳记监视元素
- connections_top – 最大并行连接数监视元素
- conn_complete_time – 连接请求完成时间戳记监视元素
- prev_uow_stop_time – 上一个工作单元完成时间戳记监视元素
- uow_start_time – 工作单元开始时间戳记监视元素
- uow_stop_time – 工作单元停止时间戳记监视元素
- uow_elapsed_time – 最近的工作单元耗用时间监视元素
- uow_comp_status – 工作单元完成状态监视元素
- uow_status – 工作单元状态监视元素
- appl_idle_time – 应用程序空闲时间监视元素
- DB2 代理程序信息监视元素

agent_id – 应用程序句柄（代理程序标识）

元素标识 agent_id

元素类型 信息

表 55. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 56. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-
语句	event_stmt	-
语句	event_subsection	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 应用程序在系统范围内的**唯一**标识。在单分区数据库中，此标识由一个 16 位计数器构成。在多分区数据库中，此标识由与 16 位计数器并置的坐标分区号构成。此外，此标识在应用程序可能进行辅助连接的每个分区上将会是一样的。

用法 应用程序句柄可用来唯一标识活动应用程序（应用程序句柄与代理程序标识同步）。

注: 根据您使用的 DB2 的版本，*agent_id* 监视元素会有不同的行为。从版本为 SQLM_DBMON_VERSION1 或 SQLM_DBMON_VERSION2 的 DB2 获取快照并发送至 DB2（版本 5 或更高版本）数据库时，返回的 *agent_id* 不能用作应用程序标识，而是作为应用程序提供服务的代理程序的 *agent_pid*。在这些情况下，仍然会返回 *agent_id* 以实现向后兼容性，但 DB2 数据库服务器内部不再将该值识别为 *agent_id*。

此值可用作需要代理程序标识的 GET SNAPSHOT 命令的输入。

读取事件跟踪时，它可用来将事件记录与给定应用程序相匹配。

它还可用作 FORCE APPLICATION 命令或 API 的输入。在多节点系统上，可从应用程序具有连接的任何节点发出此命令。它的影响是全局性的。

appl_status － 应用程序状态

元素标识 appl_status

元素类型 信息

表 57. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁	appl_lock_list	基本

表 58. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

描述 应用程序的当前状态。

用法 此元素可以帮助您诊断潜在的应用程序问题。此字段的值是:

API 常量	描述
SQLM_CONNECTPEND	**数据库连接暂挂:** 应用程序已启动数据库连接请求，但该请求尚未完成。

API 常量	描述
SQLM_CONNECTED	**数据库连接已完成**：应用程序已启动数据库连接请求，该请求已完成。
SQLM_UOWEXEC	**工作单元正在执行**：数据库管理器正在代表工作单元执行请求。
SQLM_UOWWAIT	**工作单元正在等待**：数据库管理器代表应用程序中的工作单元正在等待。此状态通常表示系统正在执行应用程序代码。
SQLM_LOCKWAIT	**等待锁定**：工作单元正在等待锁定。获取锁定之后，状态将恢复为其先前值。
SQLM_COMMIT_ACT	**正在落实**：工作单元正在落实对其数据库所作的更改。
SQLM_ROLLBACK_ACT	**正在回滚**：工作单元正在回滚对其数据库所作的更改。
SQLM_RECOMP	**正在重新编译**：数据库管理器正在为应用程序重新编译（即重新绑定）方案。
SQLM_COMP	**正在编译**：数据库管理器正在为应用程序编译 SQL 语句或者预编译方案。
SQLM_INTR	**请求已中断**：正在处理请求中断。
SQLM_DISCONNECTPEND	**数据库断开连接暂挂**：应用程序已启动数据库断开连接命令，但该命令尚未完成执行。可能是应用程序未显式执行数据库断开连接命令。如果应用程序在未执行断开连接命令的情况下结束，数据库管理器就会断开与数据库的连接。
SQLM_DECOUPLED	**已经与代理程序解耦**：当前没有与应用程序相关联的代理程序。这是一种正常状态。当连接集中器处于启用状态时，没有专用的协调代理程序，因此可以在协调程序分区中将应用程序解耦。在非集中器环境中，由于始终有专用的协调代理程序，所以无法在协调程序分区中将应用程序解耦。
SQLM_TPREP	**事务已准备好**：该工作单元是已进入两阶段落实协议准备阶段的全局事务的一部分。
SQLM_THCOMT	**事务已试探性落实**：该工作单元是已试探性落实的全局事务的一部分。
SQLM_THABRT	**事务已试探性回滚**：该工作单元是已试探性回滚的全局事务的一部分。
SQLM_TEND	**事务已结束**：该工作单元是已结束但尚未进入两阶段落实协议准备阶段的全局事务的一部分。
SQLM_CREATE_DB	**正在创建数据库**：代理程序已启动了数据库创建请求，该请求尚未完成。
SQLM_RESTART	**正在重新启动数据库**：应用程序正在重新启动数据库以执行崩溃恢复。
SQLM_RESTORE	**正在恢复数据库**：应用程序正在将备份映像恢复到数据库。
SQLM_BACKUP	**正在备份数据库**：应用程序正在执行数据库备份。
SQLM_LOAD	**数据快速装入**：应用程序正在执行“快速装入”以便将数据装入到数据库中。

API 常量	描述
SQLM_UNLOAD	**数据快速卸装**: 应用程序正在执行“快速卸装”以便从数据库中卸装数据。
SQLM_IOERROR_WAIT	**等待以禁用表空间**: 应用程序检测到 I/O 错误，并且正在尝试禁用特定表空间。应用程序必须先等待对该表空间执行的所有其他活动事务完成，然后才能禁用该表空间。
SQLM_QUIESCE_TABLESPACE	**正在停顿表空间**: 应用程序正在执行停顿表空间请求。
SQLM_WAITFOR_REMOTE	**等待远程分区**: 应用程序正在等待来自分区数据库实例中的远程分区的响应。
SQLM_REMOTE_RQST	**远程请求暂挂**: 应用程序正在等待来自联合数据源的结果。
SQLM_ROLLBACK_TO_SAVEPOINT	**回滚到保存点**: 应用程序正在回滚到保存点。

相关参考:

- 第 162 页的『status_change_time – 应用程序状态更改时间』
- 第 366 页的『stmt_operation/operation – 语句操作』

codepage_id – 应用程序使用的代码页的标识

元素标识 codepage_id

元素类型 信息

表 59. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 60. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-
连接	event_connheader	-

描述 代码页标识。

用法 对于快照监视器数据，这是启动被监视应用程序的分区的代码页。此标识可用于远程应用程序的问题确定。可以使用此信息来确保应用程序代码页与数据库代码页（或者，对于 DRDA® 主机数据库则为主机 CCSID）之间的数据转换是受支持的。有关受支持代码页的信息，请参阅*管理指南*。

对于事件监视器数据，这是对其收集事件数据的数据库的代码页。可使用此元素来确定事件监视器应用程序是否在与数据库使用的代码页不同的代码页中运行。事件监视器写下的数据使用数据库代码页。如果事件监视器应用程序使用另一代码页，则可能需要执行一些字符转换来使数据可读。

status_change_time － 应用程序状态更改时间

元素标识 status_change_time

元素类型 时间戳记

表 61. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	工作单元，时间戳记
锁	appl_lock_list	工作单元，时间戳记
DCS 应用程序	dcs_appl_info	工作单元，时间戳记

描述 应用程序进入当前状态的日期和时间。

用法 此元素允许您确定应用程序进入当前状态后所经历的时间。如果应用程序已处于同一状态很长一段时间，则可能指示存在问题。

相关参考:

- 第 159 页的『appl_status － 应用程序状态』

appl_id_oldest_xact － 带有最旧事务的应用程序

元素标识 appl_id_oldest_xact

元素类型 信息

表 62. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 具有最旧事务的应用程序的标识（对应于应用程序快照中的 *agent_id* 值）。

用法 此元素可帮助您确定具有最旧活动事务的应用程序。可强制此应用程序释放日志空间。如果它占用了大量日志空间，则应检查应用程序以确定是否可以修改它以提高执行落实操作的频率。

有时没有事务停止记录或者最旧的事务没有应用程序标识（例如，不确定事务或不活动事务）。在这类情况下，数据流中不会返回此应用程序的标识。

相关参考:

- 第 305 页的『agent_id_holding_lock － 挂起锁定的代理程序标识』
- 第 288 页的『deadlocks － 检测到的死锁数』

smallest_log_avail_node － 带有最少可用日志空间的节点

元素标识 smallest_log_avail_node

元素类型 信息

表 63. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 此元素指示带有最少可用日志空间量（以字节计）的节点并且仅对全局快照返回。

用法 将此元素与 appl_id_oldest_xact 一起使用以确保数据库有足够的日志空间可用。在全局快照中，appl_id_oldest_xact、total_log_used 和 total_log_available 对应于此节点上的值。

相关参考:

- 第 162 页的『appl_id_oldest_xact － 带有最旧事务的应用程序』
- 第 278 页的『total_log_used － 使用的总日志空间』
- 第 278 页的『total_log_available － 可用的总日志量』

appl_name － 应用程序名称

元素标识 appl_name

元素类型 信息

表 64. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 65. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 正在客户机上运行的并且数据库或 DB2 Connect 服务器所知道的应用程序的名称。

用法 此元素可与 *appl_id* 配合使用以使数据项与应用程序相关。

在客户机／服务器环境中建立数据库连接时此名称将从客户机传递至服务器。CLI 应用程序可通过对 SQLSetConnectAttr 的调用来设置 SQL_ATTR_INFO_PROGRAMNAME 属性。如果 SQL_ATTR_INFO_PROGRAMNAME 是在建立与服务器的连接前设置的，则指定的值将覆盖实际客户机应用程序名称并且成为 *appl_name* 监视元素中显示的值。

如果客户机应用程序代码页与运行 数据库系统监视器 的代码页不同，可使用 *codepage_id* 来帮助转换 *appl_name*。

相关参考:

- 第 163 页的『appl_id － 应用程序标识』
- 第 161 页的『codepage_id － 应用程序使用的代码页的标识』

appl_id － 应用程序标识

元素标识 appl_id

元素类型 信息

表 66. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
DCS 应用程序	dcs_appl_info	基本
锁	appl_lock_list	基本

表 67. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
连接	event_connheader	-
语句	event_stmt	-
事务	event_xact	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 此标识是在应用程序连接至数据库管理器上的数据库或 DDCS 接收到连接至 DRDA 数据库的请求时生成的。

用法 此标识在客户机和服务器上都是已知的，所以可使用它来使应用程序的客户机部分与服务器部分相关。对于 DDCS 应用程序，则还需要使用 outbound_appl_id 来使应用程序的客户机部分与服务器部分相关。

此标识在网络上是唯一的。应用程序标识有不同的格式，这取决于运行数据库管理器和／或 DDCS 的客户机与服务器之间的通信协议。每种格式由用逗号隔开的三个部分构成。

1. APPC

   格式 `Network.LU Name.Application instance`

   示例 `CAIBMTOR.OSFDBX0.930131194520`

   详细信息 此应用程序标识是实际 SNA LUWID（逻辑工作单元标识）的可显示格式，该 SNA LUWID 将在分配 APPC 对话时在网络上流动。APPC 生成的应用程序标识是通过并置网络名、LU 名和 LUWID 实例号构成的，这些应用程序标识将为客户机／服务器应用程序创建唯一标注。网络名和 LU 名的最大长度都是 8 个字符。应用程序实例对应于 12 位十进制字符的 LUWID 实例号。

2. TCP/IP

   格式 `IPAddr.Port.Application instance`

   IPv4

   示例 `G91A3955.F33A.02DD18143340`

   详细信息
   在 IPv4 中，TCP/IP 生成的应用程序标识由三个部分组成。第一个部分包含 IP 地址。它表示为 32 位数字，最大显示为 8 位十六进制字符。第二部分

包含端口号，表示为 4 位十六进制字符。第三部分包含此应用程序的实例的唯一标识。

注: 当 IP 地址或端口号的十六进制版本以 0 至9 开头，则它们将分别转换为 G 至 P。例如，“0”映射为“G”，“1”映射为“H”，以此类推。

IP 地址 AC10150C.NA04.006D07064947 表示为如下所示:

- IP 地址仍为 AC10150C，它将转换为 172.16.21.12。
- 端口号为 NA04。第一个字符为“N”，它将映射为“7”。因此，端口号的十六进制为 7A04，它将转换为十进制格式 31236。

IPv6

示例

```
1111:2222:3333:4444:5555:6666:
7777:8888.65535.0123456789AB
```

详细信息

在 IPv6 中，TCP/IP 生成的应用程序标识由三个部分组成。第一个部分包含 IP 地址，它是格式为 a:b:c:d:e:f:g:h 的 39 位可读地址，a 至 h 中的每一项为 4 位十六进制数字。第二个部分为可读的 5 字节端口号。第三部分是此应用程序的实例的唯一时间戳记标识。

3. IPX/SPX

格式 `Netid.nodeid.Application instance`

示例 `C11A8E5C.400011528250.0131214645`

详细信息 IPX/SPX 生成的应用程序标识是通过并置字符网络标识（8 个十六进制字符）、节点标识（12 个十六进制字符）和应用程序实例的唯一标识构成的。应用程序实例对应于格式为 MMDDHHMMSS 的 10 位十进制字符时间戳记。

4. NetBIOS

格式 `*NETBIOS.nname.Application instance`

示例 `*NETBIOS.SBOIVIN.930131214645`

详细信息 对于非分区数据库系统，NetBIOS 应用程序标识是通过并置字符串“*NETBIOS”、客户机的数据库配置文件中定义的 nname 和此应用程序的实例的唯一标识构成的。对于分区数据库系统，NetBIOS 应用程序标识是通过并置字符串“Nxxx.etc”构成的，其中 xxx 是连接应用程序的分区。

5. 本地应用程序

格式 `*LOCAL.DB2 instance.Application instance`

示例 `*LOCAL.DB2INST1.930131235945`

详细信息 为本地应用程序生成的应用程序标识是通过并置字符串 *LOCAL、DB2 实例的名称和此应用程序的实例的唯一标识构成的。

对于多分区实例，LOCAL 将替换为 Nx，其中 x 是客户机用来连接至数据库的分区号。例如，*N2.DB2INST1.0B5A12222841。

使用 *client_protocol* 来确定连接使用的通信协议，并因此确定 *appl_id* 的格式。

相关参考:

- 第 169 页的『outbound_appl_id － 出站应用程序标识』
- 第 172 页的『client_protocol － 客户机通信协议』

sequence_no － 序号

元素标识 sequence_no

元素类型 信息

表 68. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
DCS 应用程序	dcs_appl_info	基本

表 69. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
连接	event_connheader	-
语句	event_stmt	-
事务	event_xact	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-
带有详细信息的死锁的历史纪录	event_detailed_dlcomm event_stmt_history	-
带有详细信息的死锁的历史记录值	event_detailed_dlcomm event_stmt_history	-

描述 每当工作单元结束（即 COMMIT 或 ROLLBACK 终止工作单元）时，此标识就会递增。appl_id 与 sequence_no 一起唯一地标识一个事务。

auth_id － 授权标识

元素标识 auth_id

元素类型 信息

表 70. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本

表 70. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
锁	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 71. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 调用正在监视的应用程序的用户字节授权标识。在 DB2 Connect 网关节点上，这是用户在主机上的授权标识。

用法 可使用此元素来确定调用该应用程序的人员。

相关参考:

- 第 163 页的『appl_name – 应用程序名称』

session_auth_id – 会话授权标识

元素标识 session_auth_id

元素类型 信息

表 72. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁	appl_lock_list	基本

描述 此应用程序将使用的会话的当前授权标识。

用法 可使用此元素来确定要用于预编译 SQL 语句和／或执行 SQL 语句的授权标识。

client_nname – 客户机的配置 NNAME

元素标识 client_nname

元素类型 信息

表 73. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
DCS 应用程序	dcs_appl_info	基本

表 74. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 客户机数据库分区上数据库管理器配置文件中的 *nname*。

用法 可使用此元素来标识运行该应用程序的客户机数据库分区。此元素仅适用于具有 NetBIOS LAN 环境的 Windows 环境。

相关参考:

- 第 144 页的『server_nname – 监视（服务器）数据库分区上的配置 NNAME 』

client_prdid – 客户机产品／版本标识

元素标识 client_prdid

元素类型 信息

表 75. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
DCS 应用程序	dcs_appl_info	基本

表 76. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 正在客户机上运行的产品和版本。

用法 可使用此元素来标识数据库客户机的产品和代码版本。其格式为 PPPVVRRM, 其中:

- PPP 标识产品，对于 DB2 产品为“SQL”
- VV 标识两位版本号（版本只有一位时则高位为 0）
- RR 标识两位发行版号（发行版只有一位时高位为 0）
- M 标识一位修改级别。

相关参考:

- 第 146 页的『server_prdid – 服务器产品／版本标识』

client_db_alias – 应用程序使用的数据库别名

元素标识 client_db_alias

元素类型 信息

表 77. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁	appl_lock_list	基本

表 78. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 由要连接至数据库的应用程序提供的数据库别名。

用法 此元素可用来标识应用程序正在访问的实际数据库。此名称与 *db_name* 之间的映射可通过在客户机节点和数据库管理器服务器节点上使用数据库目录来实现。

这是在发出数据库连接请求的数据库管理器中定义的别名。

此元素还可用于帮助您确定认证类型，原因是不同数据库别名可能具有不同的认证类型。

相关参考:

- 第 397 页的『last_reset － 最后复位时间戳记』
- 第 398 页的『input_db_alias － 输入数据库别名』
- 第 149 页的『db_name － 数据库名称』

host_prdid － 主机产品／版本标识

元素标识 host_prdid

元素类型 信息

表 79. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

描述 正在服务器上运行的产品和版本。

用法 用于标识 DRDA 主机数据库产品及其代码版本号。其格式为 PPPVVRRM，其中:

- PPP 标识主机 DRDA 产品
  - ARI 用于 DB2 服务器 VSE 和 VM 版
  - DSN 用于 DB2 OS/390® 和 z/OS® 版
  - QSQ 用于 DB2 UDB AS/400 版
  - SQL 用于其他 DB2 产品。
- VV 标识两位版本号（版本只有一位时则高位为 0）
- RR 标识两位发行版号（发行版只有一位时高位为 0）
- M 标识一位修改级别

outbound_appl_id － 出站应用程序标识

元素标识 outbound_appl_id

元素类型 信息

表 80. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

描述 此标识是在应用程序连接至 DRDA 主机数据库时生成的。它用来将 DB2 Connect 网关连接至主机，而 *appl_id* 用来将客户机连接至 DB2 Connect 网关。

用法 可以将此元素与 *appl_id* 一起使用，以使应用程序信息的客户机部分与服务器部分相关联。

此标识在网络上是唯一的。

当网关集中器处于打开状态，或者 DCS 应用程序不在逻辑工作单元中时，此元素将是空白的。

格式 `Network.LU Name.Application instance`

示例 `CAIBMTOR.OSFDBM0.930131194520`

详细信息 此应用程序标识是实际 SNA LUWID（逻辑工作单元标识）的可显示格式，该 SNA LUWID 将在分配 APPC 对话时在网络上流动。APPC 生成的应用程序标识是通过并置网络名、LU 名和 LUWID 实例号构成的，这些应用程序标识将为客户机／服务器应用程序创建唯一标注。网络名和 LU 名的最大长度都是 8 个字符。应用程序实例对应于 12 位十进制字符的 LUWID 实例号。

相关参考:

- 第 163 页的『appl_id － 应用程序标识』

outbound_sequence_no － 出站序号

元素标识 outbound_sequence_no

元素类型 信息

表 81. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

描述 当网关集中器处于打开状态，或者 DCS 应用程序不在逻辑工作单元中时，此元素将是空白的。

execution_id － 用户登录标识

元素标识 execution_id

元素类型 信息

表 82. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 83. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 用户在登录至操作系统时指定的标识。此标识与 auth_id 不同，auth_id 是用户在连接至数据库时指定的。

用法 可使用此元素来确定个人的操作系统用户标识，这些人正在运行正在监视的应用程序。

相关参考:

- 第 166 页的『auth_id – 授权标识』

corr_token – DRDA 关联标记

元素标识 corr_token

元素类型 信息

表 84. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本

表 85. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 DRDA AS 关联标记。

用法 DRDA 关联标记用于关联应用程序服务器与应用程序请求器之间的处理。它是出现错误时转储至日志的标识，可使用该标识来标识存在错误的转换。在某些情况下，它将成为对话的 LUWID。

如果通信未使用 DRDA，则此元素返回 *appl_id*（请参阅 appl_id）。

如果要使用数据库系统监视器 API，则注意 API 常量 SQLM_APPLID_SZ 用于定义此元素的长度。

client_pid – 客户机进程标识

元素标识 client_pid

元素类型 信息

表 86. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 87. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 建立与数据库的连接的客户机应用程序的进程标识。

用法 可使用此元素使 CPU 和 I/O 时间之类的监视器信息与客户机应用程序相关。

如果是 DRDA AS 连接，则此元素将设置为 0。

client_platform － 客户机操作平台

元素标识 client_platform

元素类型 信息

表 88. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 89. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 运行客户机应用程序的操作系统。

用法 此元素可用于远程应用程序的问题确定。可在头文件 *sqlmon.h* 中找到此字段的值。

相关参考:

- 第 147 页的『server_platform － 服务器操作系统』

client_protocol － 客户机通信协议

元素标识 client_protocol

元素类型 信息

表 90. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 91. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

描述 客户机应用程序用于与服务器通信的通信协议。

用法 此元素可用于远程应用程序的问题确定。此字段的值是:

API 常量	通信协议
SQLM_PROT_UNKNOWN	（注释 1）
SQLM_PROT_LOCAL	无（注释 2）
SQLM_PROT_APPC	APPC
SQLM_PROT_TCPIP	TCP/IP

SQLM_PROT_IPXSPX	IPX/SPX
SQLM_PROT_NETBIOS	NETBIOS

注:

1. 客户机使用未知协议进行通信。仅当将来客户机与下级服务器连接时，才返回此值。
2. 客户机与服务器在同一节点上运行，未使用任何通信协议。

territory_code － 数据库地域代码

元素标识 territory_code

元素类型 信息

表 92. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本

表 93. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-
连接	event_connheader	-

描述 对其收集监视器数据的数据库的地域代码。此监视元素先前称为 country_code。

用法 地域代码信息记录在数据库配置文件中。

对于 DRDA AS 连接，此元素将设置为 0。

appl_priority － 应用程序代理程序优先级

元素标识 appl_priority

元素类型 信息

表 94. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 95. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

描述 为此应用程序工作的代理程序的优先级。

用法 可使用此元素来检查应用程序是否以期望的优先级运行。应用程序优先级可由管理员设置。可通过控制器实用程序（**db2gov**）来更改优先级。

DB2 使用控制器来监视和更改对数据库运行的应用程序的行为。此信息用来调度应用程序和平衡系统资源。

控制器守护程序通过获取快照来收集有关应用程序的统计信息。它将对照管理对该数据库运行的应用程序的规则来检查这些统计信息。如果控制器检测到规则违例，则采取适当的操作。这些规则和操作是由您在控制器配置文件中指定的。

如果与某个规则相关联的操作将更改应用程序的优先级，则控制器将在检测到违例的分区中更改代理程序的优先级。

相关参考:

- 第 174 页的『appl_priority_type － 应用程序优先级类型』

appl_priority_type － 应用程序优先级类型

元素标识 appl_priority_type

元素类型 信息

表 96. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 97. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

描述 代表应用程序工作的代理程序的操作系统优先级类型。

用法 动态优先级由操作系统根据使用情况重新计算。静态优先级不会更改。

相关参考:

- 第 376 页的『query_cost_estimate － 查询成本估计』
- 第 173 页的『appl_priority － 应用程序代理程序优先级』

authority_lvl － 用户权限级别

元素标识 authority_lvl

元素类型 信息

表 98. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
应用程序	appl_info	基本

表 99. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

描述 授予应用程序的最高权限级别。

用法 直接或间接授权进行应用程序允许的操作。

下面的定义来自 sql.h，可用来确定显式授予用户的权限:

- SQL_SYSADM
- SQL_DBADM
- SQL_CREATETAB
- SQL_BINDADD
- SQL_CONNECT
- SQL_CREATE_EXT_RT
- SQL_CREATE_NOT_FENC
- SQL_SYSCTRL
- SQL_SYSMAINT

下面的定义来自 sql.h，可用来确定从组或公用继承的间接权限:

- SQL_SYSADM_GRP
- SQL_DBADM_GRP
- SQL_CREATETAB_GRP
- SQL_BINDADD_GRP
- SQL_CONNECT_GRP
- SQL_CREATE_EXT_RT_GRP
- SQL_CREATE_NOT_FENC_GRP
- SQL_SYSCTRL_GRP
- SQL_SYSMAINT_GRP

相关概念:

- 『权限、特权和对象所有权』（*《管理指南：实施》*）

node_number － 节点号

元素标识 node_number

元素类型 信息

表 100. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本
数据库管理器	memory_pool	基本
数据库管理器	fcm	基本
数据库管理器	fcm_node	基本
数据库管理器	utility_info	基本
数据库	detail_log	基本
缓冲池	bufferpool_nodeinfo	缓冲池
表空间	rollforward	基本
锁	lock	基本
锁	lock_wait	基本
数据库	db_sto_path_info	缓冲池

表 101. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-
死锁	lock	-
溢出记录	event_overflow	-
数据库	event_dbmemuse	-
连接	event_connmemuse	-

描述 在 *db2nodes.cfg* 文件中指定给节点的编号。

用法 此值标识当前节点号，在监视多个节点时可使用节点号。

coord_node – 协调节点

元素标识 coord_node

元素类型 信息

表 102. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 103. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

描述 在多节点系统中，这是应用程序连接至实例的节点的节点号。

用法 每个连接的应用程序都有一个协调程序节点为其提供服务。

appl_con_time – 连接请求启动时间戳记

元素标识 appl_con_time

元素类型 时间戳记

表 104. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

描述 应用程序启动连接请求的日期和时间。

用法 使用此元素来确定应用程序启动连接至数据库的请求的时间。

相关参考:

- 第 177 页的『conn_complete_time – 连接请求完成时间戳记』

connections_top – 最大并行连接数

元素标识 connections_top

元素类型 水位标记

表 105. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 106. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 自数据库激活后同时与数据库进行的连接的最大数目。

用法 可使用此元素来评估 *maxappls* 配置参数的设置（*管理指南*中有对该参数的描述）。

如果此元素的值与 *maxappls* 参数相同，则可能拒绝了某些数据库连接请求，原因是 *maxappls* 限制了允许的数据库连接数。

可使用以下公式来计算获取快照时的当前连接数:

```
rem_cons_in + local_cons
```

相关参考:

- 第 184 页的『rem_cons_in – 与数据库管理器的远程连接数』
- 第 185 页的『local_cons – 本地连接数』

conn_complete_time – 连接请求完成时间戳记

元素标识 conn_complete_time

元素类型 时间戳记

表 107. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

描述 对连接请求授权的日期和时间。

用法 使用此元素来确定对数据库的连接请求授权的时间。

相关参考:

- 第 176 页的『appl_con_time – 连接请求启动时间戳记』

prev_uow_stop_time – 上一个工作单元完成时间戳记

元素标识 prev_uow_stop_time

元素类型 时间戳记

表 108. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

描述 这是工作单元的完成时间。

用法 可将此元素与 *uow_stop_time* 配合使用来计算 COMMIT/ROLLBACK 点之间的总耗用时间，还可将其与 *uow_start_time* 配合使用来计算在工作单元之间的应用程序上耗用的时间。以下其中一个时间:

- 对于当前在工作单元中的应用程序，这是最新工作单元的完成时间。
- 对于当前不在工作单元中的应用程序（该应用程序已完成某个工作单元，但尚未启动新的工作单元），这是刚刚完成的工作单元之前完成的最后一个工作单元的停止时间。刚刚完成的工作单元的时间用 uow_stop_time 指示。
- 对于第一个工作单元中的应用程序，这是数据库连接请求的完成时间。

相关参考:

- 第 178 页的『uow_start_time － 工作单元开始时间戳记』
- 第 179 页的『uow_stop_time － 工作单元停止时间戳记』
- 第 177 页的『conn_complete_time － 连接请求完成时间戳记』

uow_start_time － 工作单元开始时间戳记

元素标识 uow_start_time

元素类型 时间戳记

表 109. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

描述 工作单元第一次需要数据库资源的日期和时间。

用法 此资源需求出现在该工作单元的第一个 SQL 语句执行中:

- 对于第一个工作单元，这是 *conn_complete_time* 之后的第一个数据库请求（SQL 语句执行）的时间。
- 对于后续工作单元，这是上一个 COMMIT 或 ROLLBACK 之后第一个数据库请求（SQL 语句执行）的时间。

注: *SQL Reference* 将工作单元的边界定义为 COMMIT 或 ROLLBACK 点。

数据库系统监视器排除 COMMIT/ROLLBACK 与工作单元定义中的下一个 SQL 语句之间所花的时间。此量度方法反映数据库管理器在处理数据库请求时所花的时间，并且将此时间与该工作单元的第一个 SQL 语句之间的应用程序逻辑所花的时间隔开。工作单元耗用时间包括在工作单元内的 SQL 语句之间运行应用程序逻辑所花的时间。

可将此元素与 *uow_stop_time* 配合使用来计算工作单元的总耗用时间，并与 *prev_uow_stop_time* 配合使用来计算在工作单元之间的应用程序上耗用的时间。

可使用 *uow_stop_time* 和 *prev_uow_stop_time* 来计算的 SQL Reference 定义的工作单元的耗用时间。

相关参考:

- 第 179 页的『uow_stop_time － 工作单元停止时间戳记』
- 第 177 页的『prev_uow_stop_time － 上一个工作单元完成时间戳记』

• 第 177 页的『conn_complete_time － 连接请求完成时间戳记』

uow_stop_time － 工作单元停止时间戳记

元素标识 uow_stop_time

元素类型 时间戳记

表 110. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

描述 最新工作单元的完成日期和时间，该工作单元将在落实或回滚数据库更改时完成。

用法 可将此元素与 *prev_uow_stop_time* 配合使用来计算 COMMIT/ROLLBACK 点之间的总耗用时间，还可将其与 *uow_start_time* 配合使用来计算最新工作单元所耗用的时间。

时间戳记内容的设置将为如下所示:

- 如果应用程序完成了一个工作单元并且尚未启动新的工作单元（就像 *uow_start_time* 中定义的那样），此元素将是有效的非零时间戳记。
- 如果应用程序目前在执行工作单元，则此元素将包含零。
- 当应用程序第一次连接至数据库时，此元素将设置为 *conn_complete_time*。

启动新工作单元时，此元素的内容将移至 *prev_uow_stop_time*。

相关参考:

• 第 178 页的『uow_start_time － 工作单元开始时间戳记』
• 第 177 页的『prev_uow_stop_time － 上一个工作单元完成时间戳记』
• 第 177 页的『conn_complete_time － 连接请求完成时间戳记』

uow_elapsed_time － 最新工作单元耗用时间

元素标识 uow_elapsed_time

元素类型 时间

表 111. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

描述 最新完成的工作单元耗用的执行时间。

用法 将此元素用作完成工作单元所花时间的指示符。

相关参考:

• 第 445 页的『gw_comm_errors － 通信错误数』
• 第 446 页的『gw_comm_error_time － 通信错误时间』

uow_comp_status － 工作单元完成状态

元素标识 uow_comp_status

元素类型 信息

表 112. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元
DCS 应用程序	dcs_appl	基本

表 113. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	-

描述 工作单元的状态以及停止方式。

用法 可使用此元素来确定工作单元是否因为死锁或异常终止而结束。它可能已经:
- 因为落实语句而落实
- 因为回滚语句而回滚
- 因为死锁而回滚
- 因为异常终止而回滚
- 在正常应用程序终止时落实。
- 因为对正在运行的工作单元执行 FLUSH EVENT MONITOR 命令而处于未知状态。

注: API 用户应参考包含数据库系统监视器常量的定义的头文件（*sqlmon.h*）。

uow_status － 工作单元状态

元素标识 uow_status

元素类型 信息

表 114. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	-

描述 工作单元的状态。

用法 可使用此元素来确定工作单元的状态。API 用户应参考包含数据库系统监视器常量定义的 sqlmon.h 头文件。

appl_idle_time － 应用程序空闲时间

元素标识 appl_idle_time

元素类型 信息

表 115. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句

表 115. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句

描述 应用程序对服务器发出任何请求后经历的秒数。这包括未终止事务的应用程序，如未发出落实或回滚的应用程序。

用法 此信息可用来实现强制用户空闲指定秒数的应用程序。

相关参考:

- 第 376 页的『query_cost_estimate － 查询成本估计』
- 第 173 页的『appl_priority － 应用程序代理程序优先级』
- 第 174 页的『appl_priority_type － 应用程序优先级类型』

data_partition_id － 数据分区标识监视元素

元素标识 data_partition_id

元素类型 信息

表 116. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本
锁	lock	锁定
锁	lock_wait	锁定

表 117. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-
死锁	lock	-

描述 与返回的信息相对应的数据分区的标识。

用法 此元素仅适用于分区表。

当返回锁定级别信息时，值 -1 代表一个控制对整个表进行访问的锁定。对于非分区表来说，快照中不存在此元素。

相关任务:

- 第 26 页的『使用快照监视器数据来监视分区表的重组』

DB2 代理程序信息

DB2 代理程序信息监视元素

下列数据库系统监视元素提供有关代理程序的信息:

- agent_pid － 进程或线程标识监视元素

- coord_agent_pid – 协调程序代理程序监视元素

agent_pid – 进程或线程标识

元素标识	agent_pid
元素类型	信息

表 118. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	agent	语句

描述 DB2 代理程序的进程标识（UNIX 系统）或线程标识（Windows 系统）。

用法 可以使用此元素来将数据库系统监视器信息链接至其他诊断信息源，如系统跟踪。还可使用它来监视为数据库应用程序工作的代理程序使用系统资源的方式。

相关参考:

- 第 182 页的『coord_agent_pid – 协调程序代理程序』

coord_agent_pid – 协调程序代理程序

元素标识	coord_agent_pid
元素类型	信息

表 119. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本

描述 应用程序的协调程序代理程序的进程标识（UNIX 系统）或线程标识（Windows 系统）。

用法 可以使用此元素来将数据库系统监视器信息链接至其他诊断信息源，如系统跟踪。

相关参考:

- 第 182 页的『agent_pid – 进程或线程标识』

数据库管理器配置

数据库管理器配置监视元素

下列元素提供数据库管理器配置信息。

- 代理程序和连接监视元素
- 内存池监视元素
- 排序监视元素
- 散列连接监视元素
- 快速通信管理器监视元素
- 实用程序监视元素实用程序监视元素

代理程序和连接

代理程序和连接监视元素

代理程序是处理客户机应用程序提出的请求的进程或线程。每个连接的应用程序恰好由一个**协调代理程序**及（可能）一组从属代理程序或**子代理程序**处理。子代理程序用于分区数据库和 SMP 机器上的并行 SQL 处理。代理程序的分类如下所示:

- **协调代理程序**

  这是本地或远程应用程序连接的初始代理程序。每个数据库连接或实例连接都有一个专用协调代理程序。每个分区的最大协调代理程序数由 *max_coordagents* 配置参数控制。

- **子代理程序**

  在分区数据库中，协调代理程序可添加其他代理程序以加快 SQL 处理速度。从代理程序池选择子代理程序，并在子代理程序完成工作后将其返回至代理程序池。代理程序池的大小由 *num_poolagents* 配置参数控制。

- **关联代理程序**

  为应用程序工作的协调程序或子代理程序与该应用程序相关联。完成应用程序的工作后，它将作为关联代理程序进入代理程序池。如果应用程序尝试完成更多任务，DB2 将搜索代理程序池以获取已经与该应用程序相关联的代理程序并将任务指定给它。如果找不到任何代理程序，则 DB2 将尝试获取以下一种代理程序来满足请求:

  1. 选择未与应用程序关联的空闲代理程序。
  2. 创建代理程序（如果空闲代理程序不可用）。
  3. 查找与另一应用程序相关联的代理程序。例如，如果因为已达到 *maxagents* 而不能创建代理程序，则 DB2 将尝试获取与另一应用程序相关联的空闲代理程序。它又被称为**失窃代理程序**。

- **首要代理程序**

  DRDA 连接池中的一个网关代理程序，该代理程序连接至 DRDA 数据库以便将来在远程数据库上工作。

maxagents 配置参数定义实例可存在的代理程序的最大数目（不管类型如何）。*maxagents* 值不创建任何代理程序。在 DB2START 时在代理程序池中创建的初始代理程序数由 *num_initagents* 配置参数确定。

假定没有空闲代理程序，除非达到 *max_coordagents*，否则每个连接会创建一个新的代理程序。如果未使用子代理程序，则 *max_coordagents* 等于 *maxagents*。如果使用子代理程序，则协调代理程序和子代理程序的一些组合可能达到 *maxagents*。

对代理程序分配任务时，它将尝试获取标记或许可权以处理事务。数据库管理器使用 *maxcagents* 配置参数来控制可用的标记数。如果标记不可用，代理程序必须一直等到有可用的标记时才能处理请求的任务。这允许您使用 *maxcagents* 来控制负载、当前并行执行的事务数或服务器句柄。

下列元素提供代理程序和连接信息:
- rem_cons_in － 与数据库管理器的远程连接数监视元素
- rem_cons_in_exec － 在数据库管理器中执行的远程连接数监视元素

- local_cons – 本地连接数监视元素
- local_cons_in_exec – 在数据库管理器中执行的本地连接数监视元素
- con_local_dbases – 带有当前连接的本地数据库数监视元素
- total_cons – 数据库激活以后的连接数监视元素
- appls_cur_cons – 当前连接的应用程序数监视元素
- appls_in_db2 – 当前在数据库中执行的应用程序数监视元素
- agents_registered – 注册的代理程序数监视元素
- agents_waiting_on_token – 正在等待标记的代理程序数监视元素
- agents_registered_top – 注册的最大代理程序数监视元素
- agents_waiting_top – 正在等待的最大代理程序数监视元素
- idle_agents – 空闲代理程序数监视元素
- agents_from_pool – 从池中分配的代理程序数监视元素
- agents_created_empty_pool – 由于空的代理程序池而创建的代理程序数监视元素
- coord_agents_top – 最大协调代理程序数监视元素
- agents_stolen – 失窃代理程序数监视元素
- associated_agents_top – 最大相关代理程序数监视元素
- comm_private_mem – 已落实的专用内存监视元素
- total_sec_cons – 辅助连接数监视元素
- num_assoc_agents – 关联的代理程序数监视元素
- max_agent_overflows – 最大代理程序溢出数监视元素
- num_gw_conn_switches – 连接切换数监视元素

rem_cons_in – 与数据库管理器的远程连接数

元素标识 rem_cons_in

元素类型 标尺

表 120. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 从远程客户机启动的与正在监视的数据库管理器实例的连接的当前数目。

用法 显示此实例中从远程客户机至数据库的连接的数目。此值经常更改，所以可能需要在延长时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。此数目不包括从与数据库管理器相同的实例启动的应用程序。

与 local_cons 监视元素一起使用时，这些元素可帮助您调整 *max_coordagents* 配置参数的设置（*管理指南*中有对该参数的描述）。

相关参考:

- 第 184 页的『rem_cons_in_exec – 在数据库管理器中执行的远程连接数』
- 第 185 页的『local_cons – 本地连接数』
- 第 185 页的『local_cons_in_exec – 在数据库管理器中执行的本地连接数』

rem_cons_in_exec – 在数据库管理器中执行的远程连接数

元素标识 rem_cons_in_exec

元素类型 标尺

表 121. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 当前连接至数据库，并且正在处理要监视的数据库管理器实例中的工作单元的远程应用程序数目。

用法 此数目可帮助您确定数据库管理器上进行的并行处理的级别。此值经常更改，所以可能需要在延长时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。此数目不包括从与数据库管理器相同的实例启动的应用程序。

与 local_cons_in_exec 监视元素一起使用时，此元素可帮助您调整 *maxcagents* 配置参数的设置（*管理指南*中有对该参数的描述）。

相关参考:

- 第 184 页的『rem_cons_in － 与数据库管理器的远程连接数』
- 第 185 页的『local_cons － 本地连接数』
- 第 185 页的『local_cons_in_exec － 在数据库管理器中执行的本地连接数』

local_cons － 本地连接数

元素标识 local_cons

元素类型 标尺

表 122. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 当前连接至正在监视的数据库管理器实例中的数据库的本地应用程序的数目。

用法 此数目可帮助您确定数据库管理器中进行的并行处理的级别。此值经常更改，所以可能需要在延长时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。

此数目仅包括从与数据库管理器相同的实例启动的应用程序。 这些应用程序将进行连接，但它们可能执行也可能不执行数据库中的工作单元。

与 rem_cons_in 监视元素一起使用时，此元素可帮助您调整 *maxagents* 配置参数的设置（*管理指南*中有对该参数的描述）。

相关参考:

- 第 184 页的『rem_cons_in － 与数据库管理器的远程连接数』
- 第 184 页的『rem_cons_in_exec － 在数据库管理器中执行的远程连接数』
- 第 185 页的『local_cons_in_exec － 在数据库管理器中执行的本地连接数』

local_cons_in_exec － 在数据库管理器中执行的本地连接数

元素标识 local_cons_in_exec

元素类型 标尺

表 123. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 当前连接至正在监视的数据库管理器实例中的数据库并且正在处理工作单元的本地应用程序的数目。

用法 此数目可帮助您确定数据库管理器中进行的并行处理的级别。此值经常更改，所以可能需要在延长时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。此数目仅包括从与数据库管理器相同的实例启动的应用程序。

与 rem_cons_in_exec 监视元素一起使用时，此元素可帮助您调整 *maxcagents* 配置参数的设置（*管理指南*中有对该参数的描述）。

相关参考:

- 第 184 页的『rem_cons_in – 与数据库管理器的远程连接数』
- 第 184 页的『rem_cons_in_exec – 在数据库管理器中执行的远程连接数』
- 第 185 页的『local_cons – 本地连接数』

con_local_dbases – 带有当前连接的本地数据库数

元素标识 con_local_dbases

元素类型 标尺

表 124. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 连接了应用程序的本地数据库的数目。

用法 此值指示在数据库级别收集数据时您可以期望的数据库信息记录数。

应用程序可在本地或远程运行，并且可能执行也可能不执行数据库管理器中的工作单元。

total_cons – 数据库激活以后的连接数

元素标识 total_cons

元素类型 计数器

表 125. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本

可将快照监视的计数器复位。

表 126. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 指示第一次连接、激活或上一次复位（协调程序代理程序）后与数据库的连接的数目。

用法 可将此元素与 db_conn_time 和 db2start_time 监视元素配合使用以计算应用程序与数据库进行连接的频率。

如果连接频率很低，则您可能想要在连接任何其他应用程序之前使用 ACTIVATE DATABASE 命令显式激活数据库，这是因为第一个与数据库的连接存在其他开销（例如，初始缓冲池分配）。这将导致后续连接以较高的频率进行处理。

注: 复位此元素时，其值将设置为当前连接的应用程序数而不是设置为零。

相关参考:

- 第 150 页的『db_conn_time － 数据库激活时间戳记』
- 第 187 页的『appls_cur_cons － 当前连接的应用程序数』
- 第 187 页的『appls_in_db2 － 当前在数据库中执行的应用程序数』
- 第 193 页的『total_sec_cons － 辅助连接数』

appls_cur_cons － 当前连接的应用程序数

元素标识 appls_cur_cons

元素类型 标尺

表 127. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
锁	db_lock_list	基本

描述 指示当前连接至数据库的应用程序的数目。

用法 可使用此元素来帮助您了解数据库内的活动级别以及正在使用的系统资源量。

它可以帮助您调整 *maxappls* 和 *max_coordagents* 配置参数的设置（它们是在*管理指南*中描述的）。例如，它的值总是与 *maxappls* 相同，您可能想要提高 *maxappls* 的值。有关更多信息，请参阅 *rem_cons_in* 和 *local_cons* 监视元素。

相关参考:

- 第 187 页的『appls_in_db2 － 当前在数据库中执行的应用程序数』
- 第 186 页的『total_cons － 数据库激活以后的连接数』
- 第 184 页的『rem_cons_in － 与数据库管理器的远程连接数』
- 第 185 页的『local_cons － 本地连接数』

appls_in_db2 － 当前在数据库中执行的应用程序数

元素标识 appls_in_db2

元素类型 标尺

表 128. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 指示当前连接至数据库并且数据库管理器正对其处理请求的应用程序的数目。

用法 可使用此元素来了解连接至此数据库的应用程序正在使用的数据库管理器代理程序标记数目。如果 *rem_cons_in_exec* 与 *local_cons_in_exec* 的和等于 *maxcagents* 配置参数的值，则您可能想要增加该参数的值（*管理指南*中有对该参数的描述）。

相关参考:

- 第 187 页的『appls_cur_cons – 当前连接的应用程序数』
- 第 186 页的『total_cons – 数据库激活以后的连接数』
- 第 184 页的『rem_cons_in_exec – 在数据库管理器中执行的远程连接数』
- 第 185 页的『local_cons_in_exec – 在数据库管理器中执行的本地连接数』
- 第 303 页的『locks_waiting – 等待锁定的当前代理程序数』

agents_registered – 注册的代理程序数

元素标识 agents_registered

元素类型 标尺

表 129. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 正在监视的数据库管理器实例中的已注册代理程序的数目（协调程序代理程序和子代理程序）。

用法 可使用此元素来帮助您评估 *maxagents* 配置参数的设置。

相关参考:

- 第 189 页的『agents_registered_top – 最大已注册代理程序数』

agents_waiting_on_token – 正在等待标记的代理程序数

元素标识 agents_waiting_on_token

元素类型 标尺

表 130. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 等待标记以便在数据库管理器中执行事务的代理程序的数目。

用法 可使用此元素来帮助您评估 *maxcagents* 配置参数的设置。

每个应用程序都有一个专用协调程序代理程序来处理数据库管理器中的数据库请求。每个代理程序都必须获取标记才能执行事务。配置参数 *maxcagents* 限制可执行数据库管理器事务的代理程序的最大数目。有关此参数的更多信息，请参阅*管理指南*。

相关参考:

- 第 188 页的『agents_registered － 注册的代理程序数』

agents_registered_top － 最大已注册代理程序数

元素标识 agents_registered_top

元素类型 水位标记

表 131. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 数据库管理器自启动后曾经同时注册的代理程序（协调代理程序和子代理程序）的最大数目。

用法 可使用此元素来帮助您评估 *maxagents* 配置参数的设置（*管理指南*中有对该参数的描述）。

agents_registered 将记录获取快照时注册的代理程序数。

相关参考:

- 第 188 页的『agents_registered － 注册的代理程序数』
- 第 189 页的『agents_waiting_top － 正在等待的代理程序的最大数目』

agents_waiting_top － 正在等待的代理程序的最大数目

元素标识 agents_waiting_top

元素类型 水位标记

表 132. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 自数据库管理器启动后曾经同时等待标记的代理程序的最大数目。

用法 可使用此元素来帮助您评估 *maxcagents* 配置参数的设置（*管理指南*中有对该参数的描述）。

agents_waiting_on_token 将记录获取快照时等待标记的代理程序数。

如果 *maxcagents* 参数设置为其缺省值（-1），则不应有任何代理程序等待标记并且此监视元素的值应该为零。

相关参考:

- 第 188 页的『agents_waiting_on_token － 正在等待标记的代理程序数』
- 第 189 页的『agents_registered_top － 最大已注册代理程序数』

idle_agents － 空闲代理程序数

元素标识 idle_agents

元素类型 标尺

表 133. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 代理程序池中当前未分配给应用程序而“空闲”的代理程序的数目。

用法 可使用此元素来帮助设置 *num_poolagents* 配置参数。如果具有可用来处理代理程序请求的空闲代理程序，就可以改进性能。有关更多信息，请参阅*管理指南*。

相关参考:

- 第 189 页的『agents_registered_top － 最大已注册代理程序数』
- 第 189 页的『agents_waiting_top － 正在等待的代理程序的最大数目』
- 第 188 页的『agents_registered － 注册的代理程序数』

agents_from_pool － 从池中分配的代理程序数

元素标识 agents_from_pool

元素类型 计数器

表 134. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 从代理程序池中分配的代理程序数。

用法 此元素可与 *agents_created_empty_pool* 配合使用以确定因为池是空的而必须创建代理程序的频率。

如果

由于空的代理程序池而创建的代理程序数 / 从池中分配的代理程序数

的比率很高，则可能指示应该提高 *num_poolagents* 配置参数。如果比率很低，则表示 *num_poolagents* 设置得过高，池中的某些代理程序很少使用，因此浪费了系统资源。

高比率可能指示此节点的整体工作负载过高。可通过降低 *maxcagents* 配置参数指定的最大协调代理程序数或者在节点间重新分发数据来调整工作负载。

有关代理程序池大小（*num_poolagents*）和最大并行协调代理程序数（*maxcagents*）配置参数的更多信息，请参阅*管理指南*。

相关参考:

- 第 191 页的『agents_created_empty_pool － 由于空的代理程序池而创建的代理程序数』
- 第 191 页的『coord_agents_top － 最大协调代理程序数』

agents_created_empty_pool – 由于空的代理程序池而创建的代理程序数

元素标识 agents_created_empty_pool

元素类型 计数器

表 135. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 由于空的代理程序池而创建的代理程序数。它包括在 DB2 启动时创建的代理程序数（*num_initagents*）。

用法 与 agents_from_pool 一起使用来计算

由于空的代理程序池而创建的代理程序数 / 从池中分配的代理程序数

有关使用此元素的信息，请参阅 agents_from_pool。

相关参考:

- 第 190 页的『agents_from_pool – 从池中分配的代理程序数』
- 第 191 页的『coord_agents_top – 最大协调代理程序数』

coord_agents_top – 最大协调代理程序数

元素标识 coord_agents_top

元素类型 水位标记

表 136. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
数据库	dbase	基本

描述 同时工作的最大协调代理程序数。

用法 如果协调代理程序的峰值数目显示此节点的工作负载过高，可通过更改 *maxcagents* 配置参数来降低可并行执行事务的协调代理程序数。

有关最大并行协调代理程序数（*maxcagents*）配置参数的更多信息，请参阅*管理指南*。

相关参考:

- 第 190 页的『agents_from_pool – 从池中分配的代理程序数』
- 第 191 页的『agents_created_empty_pool – 由于空的代理程序池而创建的代理程序数』

agents_stolen – 失窃代理程序数

元素标识 agents_stolen

元素类型 计数器

表 137. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
应用程序	appl	基本

可将快照监视的计数器复位。

描述 从应用程序失窃的代理程序数。与应用程序相关联的空闲代理程序被重新分配到另一应用程序上工作时，代理程序就失窃了。

用法 此元素可与 *associated_agents_top* 一起使用来评估此应用程序放在系统上的负载。

如果 *agents_stolen* 很高，则考虑提高 *num_poolagents* 配置参数。

相关参考:

- 第 390 页的『num_agents – 正在处理语句的代理程序数』

associated_agents_top – 最大关联代理程序数

元素标识 associated_agents_top

元素类型 水位标记

表 138. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

描述 与此应用程序相关联的最大子代理程序数。

用法 如果子代理程序的峰值数目接近 *num_poolagents* 配置参数，则这可能指示此节点的工作负载过高。

有关代理程序池大小（*num_poolagents*）配置参数的更多信息，请参阅*管理指南*。

相关参考:

- 第 190 页的『agents_from_pool – 从池中分配的代理程序数』
- 第 191 页的『agents_created_empty_pool – 由于空的代理程序池而创建的代理程序数』

comm_private_mem – 已落实专用内存

元素标识 comm_private_mem

元素类型 标尺

表 139. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 目前数据库管理器的实例在获取快照时已落实的专用内存量。

用法 可使用此元素来帮助设置 *min_priv_mem* 配置参数（请参阅*管理指南*）以确保您

有足够的专用内存可用。将对所有平台返回此元素，但调整只能在 DB2 使用线程（如 Windows 2000）的平台上完成。

total_sec_cons － 辅助连接数

元素标识 total_sec_cons

元素类型 计数器

表 140. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 子代理程序与节点上的数据库建立的连接的数目。

用法 可将此元素与 total_cons、db_conn_time 和 db2start_time 监视元素一起使用来计算应用程序与数据库建立连接的频率。

相关参考:

- 第 186 页的『total_cons － 数据库激活以后的连接数』
- 第 144 页的『db2start_time － 启动数据库管理器时间戳记』
- 第 150 页的『db_conn_time － 数据库激活时间戳记』

num_assoc_agents － 关联代理程序数

元素标识 num_assoc_agents

元素类型 标尺

表 141. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl_info	基本

描述 在应用程序级别，此项是与应用程序相关联的子代理程序数。在数据库级别，此项是用于所有应用程序的子代理程序数。

用法 可使用此元素来帮助您评估代理程序配置参数的设置。

相关参考:

- 第 190 页的『agents_from_pool － 从池中分配的代理程序数』
- 第 191 页的『agents_created_empty_pool － 由于空的代理程序池而创建的代理程序数』
- 第 192 页的『associated_agents_top － 最大关联代理程序数』
- 第 193 页的『max_agent_overflows － 最大代理程序溢出数』

max_agent_overflows － 最大代理程序溢出数

元素标识 max_agent_overflows

元素类型 标尺

表 142. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 达到 *maxagents* 配置参数后接收到创建新代理程序的请求的次数。

用法 如果达到 *maxagents* 配置参数后仍然接收到代理程序创建请求，则可能指示此节点的工作负载太高。

有关最大代理程序数（*maxagents*）配置参数的更多信息，请参阅*管理指南*。

相关参考:

- 第 190 页的『agents_from_pool － 从池中分配的代理程序数』
- 第 191 页的『agents_created_empty_pool － 由于空的代理程序池而创建的代理程序数』
- 第 192 页的『associated_agents_top － 最大关联代理程序数』
- 第 193 页的『num_assoc_agents － 关联代理程序数』

num_gw_conn_switches － 连接交换次数

元素标识	num_gw_conn_switches
元素类型	标尺

表 143. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 代理程序池中的代理程序准备好进行连接但被窃走以与另一 DRDA 数据库配合使用的次数。

用法 使用此元素来确定是否应该增加代理程序池的大小。

相关参考:

- 第 188 页的『agents_registered － 注册的代理程序数』
- 第 193 页的『max_agent_overflows － 最大代理程序溢出数』
- 第 189 页的『agents_registered_top － 最大已注册代理程序数』
- 第 193 页的『num_assoc_agents － 关联代理程序数』
- 第 193 页的『total_sec_cons － 辅助连接数』

内存池

内存池监视元素

下列元素提供有关内存池的信息:
- pool_id － 内存池标识监视元素
- pool_secondary_id － 内存池辅助标识监视元素
- pool_cur_size － 内存池的当前大小监视元素
- pool_config_size － 配置的内存池大小监视元素
- pool_watermark － 内存池水位标记监视元素

memory_pool 数据元素的特征因为平台的不同而变化：在 Windows 系统上，数据库系统监视器不会在数据库快照中报告任何内存，而在 UNIX 系统上，则会在数据库快照中报告内存。Windows 系统将在数据库管理器快照中报告此内存，而不是在数据库快照中报告它。报告方面的不同是因为 Windows 系统与 UNIX 系统之间的底层内存体系结构差别造成的。

pool_id － 内存池标识

元素标识 pool_id

元素类型 信息

表 144. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 145. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

描述 内存池的类型。

用法 要跟踪系统内存使用情况，请将此值与 pool_max_size、pool_cur_size 和 pool_watermark 配合使用。

使用 pool_id 来标识系统监视器输出中讨论的内存池。各种内存池标识都可在 sqlmon.h 中找到。在正常操作情况下，可以使用下列每个内存池的其中一个或多个。

API 常量	描述
SQLM_HEAP_APPLICATION	应用程序堆
SQLM_HEAP_DATABASE	数据库堆
SQLM_HEAP_APPL_CONTROL	应用程序控制堆
SQLM_HEAP_LOCK_MGR	锁定管理器堆
SQLM_HEAP_UTILITY	备份／恢复／实用程序堆
SQLM_HEAP_STATISTICS	统计信息堆
SQLM_HEAP_PACKAGE_CACHE	程序包高速缓存堆
SQLM_HEAP_CAT_CACHE	目录高速缓存堆
SQLM_HEAP_DFM	DFM 堆
SQLM_HEAP_QUERY	查询堆
SQLM_HEAP_MONITOR	数据库监视器堆
SQLM_HEAP_STATEMENT	语句堆
SQLM_HEAP_FCMBP	FCMBP 堆
SQLM_HEAP_IMPORT_POOL	导入池
SQLM_HEAP_OTHER	其他内存

API 常量	描述
SQLM_HEAP_BP	缓冲池堆
SQLM_HEAP_APP_GROUP	应用程序组共享堆
SQLM_HEAP_SHARED_SORT	排序共享堆

相关参考:

- 第 196 页的『pool_cur_size – 内存池的当前大小』
- 第 197 页的『pool_config_size – 内存池的已配置大小』
- 第 198 页的『pool_watermark – 内存池水位标记』

pool_secondary_id – 内存池辅助标记

元素标识 pool_secondary_id

元素类型 信息

表 146. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 147. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

描述 一个附加标识，用于帮助确定对其返回监视器数据的内存池。

用法 与 pool_id 一起使用来确定对其返回监视器数据的内存池。pool_secondary_id 的数据仅在必要时才会出现。例如，当指示的 pool_id 是用于确定与监视器数据相关的缓冲池的缓冲池堆时，它就会出现。

创建新数据库后，它将具有缺省缓冲池 IBMDEFAULTBP，其大小将由平台确定。此缓冲池的辅助标识为“1”。除了此缓冲池及您创建的所有缓冲池之外，在缺省情况下还会创建一组系统缓冲池，每个缓冲池对应不同页大小。这些缓冲池的标识会出现在 pool_secondary_id 的快照中:

- 系统 32K 缓冲池
- 系统 16K 缓冲池
- 系统 8K 缓冲池
- 系统 4K 缓冲池

相关参考:

- 第 195 页的『pool_id – 内存池标识』

pool_cur_size – 内存池的当前大小

元素标识 pool_cur_size

元素类型 信息

表 148. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 149. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

描述 内存池的当前大小。

用法 要跟踪系统内存使用情况，请将此值与 *pool_config_size*、*pool_id* 和 *pool_watermark* 一起使用。

要查看内存池是否接近已满状态，将 *pool_config_size* 与 *pool_cur_size* 进行比较。例如，假定实用程序堆非常小。可以按一定时间间隔获取快照并查看快照输出的实用程序堆部分，以诊断这一特定问题。如果 *pool_cur_size* 的值一直接近 *pool_config_size*，则您可能要考虑增加实用程序堆的大小。

相关参考:

- 第 195 页的『pool_id – 内存池标识』
- 第 197 页的『pool_config_size – 内存池的已配置大小』
- 第 198 页的『pool_watermark – 内存池水位标记』

pool_config_size – 内存池的已配置大小

元素标识 pool_config_size

元素类型 信息

表 150. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 151. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

描述 DB2 数据库系统中的内存池的内部配置大小。

用法 要跟踪系统内存使用情况，请将此值与 *pool_cur_size*、*pool_id* 和 *pool_watermark* 一起使用。

要查看内存池是否接近已满状态，将 *pool_config_size* 与 *pool_cur_size* 进行比较。例如，假定实用程序堆非常小。可以按一定时间间隔获取快照并查看快照输出的实用程序堆部分，以诊断这一特定问题。如果需要，可能会允许 *pool_cur_size* 超过 *pool_config_size* 以避免内存不足故障。如果这种情况不常发生，则可能不需要进一步的操作。但是，如果 *pool_cur_size* 一直接近或大于 *pool_config_size*，则可能要考虑增加实用程序堆的大小。

相关参考:

- 第 195 页的『pool_id – 内存池标识』
- 第 196 页的『pool_cur_size – 内存池的当前大小』
- 第 198 页的『pool_watermark – 内存池水位标记』

pool_watermark – 内存池水位标记

元素标识 pool_watermark

元素类型 信息

表 152. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 153. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

描述 自创建内存池后内存池的最大大小。

用法 在一直运行的系统上，可将 *pool_watermark* 和 *pool_config_size* 元素一起使用来预测潜在的内存问题。

例如，按一定时间间隔获取快照（如每天），并检查 *pool_watermark* 和 *pool_config_size* 值。如果发现 *pool_watermark* 的值开始逐步接近 *pool_config_size*（预示将来可能出现内存相关问题），则可能指示应增加内存池的大小。

相关参考:

- 第 195 页的『pool_id – 内存池标识』
- 第 196 页的『pool_cur_size – 内存池的当前大小』
- 第 197 页的『pool_config_size – 内存池的已配置大小』

排序

排序监视元素

下列元素提供有关执行的数据库管理器排序工作的信息:
- sort_heap_allocated – 分配的总排序堆监视元素

- post_threshold_sorts – 超过阀值后的排序数监视元素
- piped_sorts_requested – 请求的管道式排序数监视元素
- piped_sorts_accepted – 接受的管道式排序数监视元素
- total_sorts – 总排序数监视元素
- total_sort_time – 总排序时间监视元素
- sort_overflows – 排序溢出数监视元素
- active_sorts – 活动排序数监视元素
- sort_shrheap_allocated – 当前分配的排序共享堆监视元素
- sort_shrheap_top – 排序共享堆高水位标记监视元素

sort_heap_allocated – 分配的总排序堆

元素标识 sort_heap_allocated

元素类型 标尺

表 154. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
数据库	dbase	基本

描述 排序堆空间的总分配页数，用于获取快照时处于所选级别的所有排序。

用法 为每个排序分配的内存量可以是可用排序堆大小的一部分或全部。排序堆大小是按 *sortheap* 数据库配置参数中定义的可用于每个排序的内存量。

单个应用程序可使多个排序同时处于活动状态。例如，在某些情况下带有子查询的 SELECT 语句可能导致并行排序。

可在两个级别收集信息:

- 在数据库管理器级别，它表示为数据库管理器的所有活动数据库中的所有排序分配的排序堆空间的总和
- 在数据库级别，它表示为数据库中的所有排序分配的排序堆空间的总和。

常规内存估计不包括排序堆空间。如果出现过多排序，除了需要用于运行数据库管理器的基本内存之外，还应额外添加用于排序堆的内存。通常排序堆越大，排序越有效。适当使用索引可以降低必需的排序量。

可使用在数据库管理器级别返回的信息来帮助您调整 *sheapthres* 配置参数。如果元素值大于或等于 *sheapthres*，则表示排序未达到 *sortheap* 参数定义的排序堆已满的状态。

相关参考:

- 第 201 页的『total_sorts – 总排序数』

post_threshold_sorts – 超过阈值后的排序数

元素标识 post_threshold_sorts

元素类型 计数器

表 155. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	排序

可将快照监视的计数器复位。

描述 超过排序堆阈值后请求堆的排序数。

用法 在正常条件下，数据库管理器将使用 *sortheap* 配置参数指定的值来分配排序堆。如果分配给排序堆的内存量超过排序堆阈值（*sheapthres* 配置参数），则数据库管理器将使用小于*sortheap*配置参数指定值的值来分配排序堆。

系统上每个活动排序分配内存可能导致排序占用过多系统可用内存。在达到排序堆阈值后启动的排序不能接收最优内存量来执行，但整个系统将因此而获益。通过修改排序堆阈值和排序堆大小配置参数，排序操作性能和整体系统性能可以得到改进。如果此元素的值过高，则可以:

- 提高排序堆阈值（*sheapthres*）或者
- 通过 SQL 查询更改将应用程序调整为使用数量较少范围较小的排序。

相关参考:

- 第 373 页的『stmt_sorts – 语句排序数』
- 第 203 页的『active_sorts – 活动排序数』

piped_sorts_requested – 请求的管道排序数

元素标识 piped_sorts_requested

元素类型 计数器

表 156. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器复位。

描述 请求的管道排序数。

用法 系统上每个活动排序分配内存可能导致排序占用过多可用系统内存。

排序列表堆（*sortheap*）和排序堆阈值（*sheapthres*）配置参数帮助控制用于排序操作的内存量。这些参数还将用于确定排序是否以管道方式进行。

因为管道排序可以降低磁盘 I/O，允许更多管道排序可以改进排序操作的性能并且可能改进整个系统的性能。如果对排序分配排序堆时将超过排序堆阈值，则不接受管道排序。如果遇到拒绝管道排序的情况，请参阅 *piped_sorts_accepted* 以获取更多信息。

SQL EXPLAIN 输出将显示优化器是否请求管道排序。有关管道排序和非管道排序的更多信息，请参阅*管理指南*。

相关参考:

- 第 200 页的『piped_sorts_accepted – 接受的管道排序数』
- 第 199 页的『post_threshold_sorts – 超过阈值后的排序数』

piped_sorts_accepted – 接受的管道排序数

元素标识 piped_sorts_accepted

元素类型 计数器

表 157. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器复位。

描述 接受的管道排序数。

用法 系统上每个活动排序分配内存可能导致排序占用过多可用系统内存。

如果接受的管道排序数低于请求的管道排序数，可通过调整下列配置参数中的一个或全部来改进排序性能:

- sortheap
- sheapthres

如果拒绝管道排序，则可以考虑降低排序堆或提高排序堆阈值。您应该了解其中每个选项的可能含义。如果提高排序堆阈值，则可能会保留更多内存以便分配给排序使用。这可能导致内存至磁盘的页面调度。如果降低排序堆，则可能需要进行其他的合并阶段，这可能会导致排序速度下降。

有关排序的更多信息，请参阅*管理指南*。

相关参考:

- 第 200 页的『piped_sorts_requested － 请求的管道排序数』
- 第 199 页的『post_threshold_sorts － 超过阈值后的排序数』

total_sorts － 总排序数

元素标识 total_sorts

元素类型 计数器

表 158. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 159. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-

描述 已执行的总排序数。

用法 在数据库或应用程序级别，将此值与 *sort_overflows* 配合使用来计算需要更多堆空间的排序的百分比。还可将其与 *total_sort_time* 配合使用来计算平均排序时间。

如果排序溢出数相对总排序数较小，则除非此缓冲区大小实际上增加了，否则提高排序堆大小对性能没什么影响。

在语句级别，使用此元素来标识执行大量排序的语句。如果另外进行调整以降低排序数目，这些语句将从中受益。还可使用 SQL EXPLAIN 语句来标识语句执行的排序数。有关更多信息，请参阅*管理指南*。

相关参考:

- 第 203 页的『sort_overflows － 排序溢出数』
- 第 202 页的『total_sort_time － 总排序时间』

total_sort_time － 总排序时间

元素标识 total_sort_time

元素类型 计数器

表 160. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	排序
应用程序	appl	排序
应用程序	stmt	排序
动态 SQL	dynsql	排序

可将快照监视的计数器复位。

表 161. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-

描述 已执行的所有排序的总耗用时间（以毫秒计）。

用法 在数据库或应用程序级别，将此元素与 *total_sorts* 配合使用来计算平均排序时间，它可以指示就性能而言排序是否存在问题。

在语句级别，使用此元素来标识花费大量时间进行排序的语句。如果另外进行调整以降低排序时间，这些语句将从中受益。

此计数还包括相关操作期间创建的临时表的排序时间。它提供有关一个语句、一个应用程序或访问一个数据库的所有应用程序的信息。

使用提供耗用时间的监视元素时，应考虑:

1. 耗用时间受系统负载影响，所以运行的进程越多，此耗用时间值越高。
2. 要在数据库级别计算此监视元素，数据库系统监视器会将应用程序级别时间求和。这会导致对数据库级别的耗用时间双倍计数，原因是多个应用程序进程可能同时运行。

   要在数据库级别提供有意义的数据，应将数据标准化以将其降至较低级别。例如:

   ```
   total_sort_time / total_sorts
   ```

   提供有关每个排序的平均耗用时间的信息。

相关参考:

- 第 201 页的『total_sorts － 总排序数』
- 第 203 页的『sort_overflows － 排序溢出数』

sort_overflows － 排序溢出数

元素标识 sort_overflows

元素类型 计数器

表 162. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	基本

可将快照监视的计数器复位。

表 163. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-

描述 用完排序堆并且可能需要磁盘空间以供临时存储器使用的总排序数。

用法 在数据库或应用程序级别，将此元素与 *total_sorts* 一起使用来计算必须溢出至磁盘的排序的百分比。如果此百分比很高，则您可能想要通过增加 *sortheap* 的值来调整数据库配置。

在语句级别，使用此元素来标识需要大量排序的语句。如果另外进行调整以降低所需排序量，这些语句将从中受益。

当排序溢出时，因为排序需要合并阶段并且可能需要更多 I/O（如果数据需要写至磁盘），所以可能会导致其他开销。

此元素提供有关一个语句、一个应用程序或访问一个数据库的所有应用程序的信息。

相关参考:

- 第 201 页的『total_sorts － 总排序数』

active_sorts － 活动排序数

元素标识 active_sorts

元素类型 标尺

表 164. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 数据库中当前分配了排序堆的排序数。

用法 将此值与 *sort_heap_allocated* 一起使用来确定每个排序使用的平均排序堆空间。如果 *sortheap* 配置参数实际上大于使用的平均排序堆，则您可以降低此参数的值。（有关更多详细信息，请参阅*管理指南*。）

此值包括用于相关操作期间创建的临时表的排序堆。

相关参考:

- 第 199 页的『sort_heap_allocated － 分配的总排序堆』
- 第 201 页的『total_sorts － 总排序数』

sort_heap_top － 排序专用堆高水位标记

元素标识 sort_heap_top

元素类型 水位标记

表 165. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

描述 数据库管理器范围的专用排序内存高水位标记（以 4K 页计）。

用法 此元素可用来确定是否已将 SHEAPTHRES 配置参数设置为最佳的值。例如，如果此水位标记接近或超过 SHEAPTHRES，则可能应该增大 SHEAPTHRES。这是因为，超过 SHEAPTHRES 后，专用排序操作可使用的内存就会减少，这会对系统性能产生不利影响。

sort_shrheap_allocated － 当前分配的共享排序堆

元素标识 sort_shrheap_allocated

元素类型 信息

表 166. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 数据库中分配的共享排序内存总量。

用法 此元素可用来评估共享排序内存的阈值。如果此值经常大幅高于或低于当前共享排序内存阈值，则可能应该调整阈值。

注: 如果 SHEAPTHRES_SHR 数据库配置参数为 0，则“共享排序内存阈值”由 SHEAPTHRES 数据库管理器配置参数确定。否则将由 SHEAPTHRES_SHR 的值确定。

sort_shrheap_top － 排序共享堆高水位标记

元素标识 sort_shrheap_top

元素类型 水位标记

表 167. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 数据库范围的共享排序内存高水位标记（以 4K 页计）。

用法 此元素可用来评估是否已将 SHEAPTHRES（或 SHEAPTHRES_SHR）设置为最佳的值。例如，如果这个高水位标记持续地远小于共享排序内存阈值，则有可能需要减小这个阈值，从而释放内存以供其他数据库功能使用。相反，如果这个高水位标记开始接近共享排序内存阈值，则可能表示需要增大这个阈值。由于共享排序内存阈值是硬限制，所以这一点很重要。排序内存总量达到这个阈值后，就无法启动更多的共享排序操作。

此元素和专用排序内存高水位标记还可以帮助用户确定是否需要相互独立地设置共享排序阈值和专用排序阈值。通常，如果 SHEAPTHRES_SHR 数据库配置选项值为 0，则共享排序内存阈值由 SHEAPTHRES 数据库管理器配置选项值确定。但是，如果专用排序内存高水位标记与共享排序内存高水位标记相差很大，则可能表明用户需要重设 SHEAPTHRES，并将 SHEAPTHRES_SHR 设置为基于共享排序内存高水位标记的更合适的值。

post_shrthreshold_sorts － 阈值后排序数

元素标识	post_shrthreshold_sorts
元素类型	计数器

表 168. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	排序

可将快照监视的计数器复位。

表 169. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 排序内存限量算法已限制的排序总数。内存受限排序是指获得的内存量少于排序内存管理器所请求内存量的排序。当为排序分配的内存量接近数据库配置参数 *sheapthres_shr* 设置的限制时，就会对排序进行内存限制。这种内存限制将显著减少在配置不良的系统中超过 *sheapthres_shr* 限制的次数。此元素报告的数据仅反映正在使用从共享排序堆分配的内存的排序。

相关参考:

- 第 203 页的『active_sorts － 活动排序数』
- 『sheapthres － 排序堆阈值配置参数』（*《性能指南》*）
- 『sheapthres_shr － 共享排序的排序堆阈值配置参数』（*《性能指南》*）

散列连接

散列连接监视元素

散列连接是优化器的附加选项。在比较连接中涉及的表的谓词之前，散列连接会先比较*散列码*。在散列连接中，将扫描一个表（由优化器选择）并且会将各行复制到从排序堆分配获取的内存缓冲区中。根据 Join 谓词的列计算出来的散列码，将内存缓冲区分为若干分区。通过比较散列码，将连接涉及的另一个表的行与第一个表的行相匹配。如果散列码匹配，则比较实际 Join 谓词列。

- total_hash_joins － 散列连接总数监视元素
- post_threshold_hash_joins － 散列连接阈值监视元素
- total_hash_loops － 散列循环总数监视元素
- hash_join_overflows － 散列连接溢出数监视元素
- hash_join_small_overflows － 散列连接小型溢出数监视元素

active_hash_joins － 活动散列连接数

元素标识 active_hash_joins

元素类型 计数器

表 170. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

表 171. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 当前正在运行并消耗内存的散列连接的总数。

相关参考:

- 第 207 页的『post_shrthreshold_hash_joins － 阈值后散列连接数』

total_hash_joins － 散列连接总数

元素标识 total_hash_joins

元素类型 计数器

表 172. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 173. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述　已执行的散列连接的总数。

用法　在数据库或应用程序级别，将此值与 hash_join_overflows 和 hash_join_small_overflows 配合使用，以确定适度增大排序堆是否能够使相当百分比的散列连接受益。

相关参考:

- 第 207 页的『post_threshold_hash_joins – 散列连接阈值』
- 第 208 页的『total_hash_loops – 总散列循环数』
- 第 208 页的『hash_join_overflows – 散列连接溢出数』
- 第 209 页的『hash_join_small_overflows – 散列连接小溢出数』

post_threshold_hash_joins – 散列连接阈值

元素标识　post_threshold_hash_joins

元素类型　计数器

表 174. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器复位。

描述　散列连接堆请求因为并行使用共享或专用排序堆空间而受限的总次数。

用法　如果此值很大（超过 hash_join_overflows 的 5%），则应增加排序堆阈值。

相关参考:

- 第 206 页的『total_hash_joins – 散列连接总数』
- 第 208 页的『total_hash_loops – 总散列循环数』
- 第 208 页的『hash_join_overflows – 散列连接溢出数』
- 第 209 页的『hash_join_small_overflows – 散列连接小溢出数』

post_shrthreshold_hash_joins – 阈值后散列连接数

元素标识　post_shrthreshold_hash_joins

元素类型　计数器

表 175. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

可将快照监视的计数器复位。

表 176. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述　排序内存限量算法已限制的散列连接总数。内存受限散列连接是指获得的内存量少于排序内存管理器所请求内存量的散列连接。当从共享排序堆中分配的内

存量接近数据库配置参数 *sheapthres_shr* 设置的限制时，就会对散列连接进行内存限制。这种内存限制将显著减少在配置不良的系统中超过 *sheapthres_shr* 限制的次数。此元素报告的数据仅反映正在使用从共享排序堆分配的内存的散列连接。

相关参考:

- 第 206 页的『active_hash_joins － 活动散列连接数』
- 『sheapthres － 排序堆阈值配置参数』（*《性能指南》*）
- 『sheapthres_shr － 共享排序的排序堆阈值配置参数』（*《性能指南》*）

total_hash_loops － 总散列循环数

元素标识 total_hash_loops

元素类型 计数器

表 177. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 178. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 散列连接的单个分区大于可用排序堆空间的总次数。

用法 此元素的值指示散列连接的执行没有效率。它可能指示排序堆大小太小或者排序堆阈值太小。将此值与其他散列连接变量一起使用来调整排序堆大小（*sortheap*）和排序堆阈值（*sheapthres*）配置参数。

相关参考:

- 第 206 页的『total_hash_joins － 散列连接总数』
- 第 207 页的『post_threshold_hash_joins － 散列连接阈值』
- 第 208 页的『hash_join_overflows － 散列连接溢出数』
- 第 209 页的『hash_join_small_overflows － 散列连接小溢出数』

hash_join_overflows － 散列连接溢出数

元素标识 hash_join_overflows

元素类型 计数器

表 179. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 180. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 散列连接数据超过可用排序堆空间的次数。

用法 在数据库级别，如果 hash_join_small_overflows 的值大于此 hash_join_overflows 的 10%，则应该考虑增加排序堆大小。应用程序级别的值可用于评估各个应用程序的散列连接性能。

相关参考:

- 第 206 页的『total_hash_joins – 散列连接总数』
- 第 207 页的『post_threshold_hash_joins – 散列连接阈值』
- 第 208 页的『total_hash_loops – 总散列循环数』
- 第 209 页的『hash_join_small_overflows – 散列连接小溢出数』

hash_join_small_overflows – 散列连接小溢出数

元素标识	hash_join_small_overflows
元素类型	计数器

表 181. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 182. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 散列连接数据超过可用排序堆空间的部分小于 10% 的次数。

用法 如果此值和 hash_join_overflows 很高，则应考虑增加排序堆阈值。如果此值超过 hash_join_overflows 的 10%，则应考虑增加排序堆大小。

相关参考:

- 第 206 页的『total_hash_joins – 散列连接总数』
- 第 207 页的『post_threshold_hash_joins – 散列连接阈值』
- 第 208 页的『total_hash_loops – 总散列循环数』
- 第 208 页的『hash_join_overflows – 散列连接溢出数』

快速通信管理器

快速通信管理器监视元素

下列数据库系统监视器元素提供有关快速通信管理器（FCM）的信息:
- buff_free – 当前可用的 FCM 缓冲区监视元素
- buff_free_bottom – 可用的最少 FCM 缓冲区监视元素
- ch_free – 当前可用的通道数监视元素ch_free – 当前可用的通道数监视元素
- ch_free_bottom – 最低可用通道数监视元素ch_free_bottom – 最低可用通道数监视元素
- connection_status – 连接状态监视元素
- total_buffers_sent – 发送的总 FCM 缓冲区数监视元素
- total_buffers_rcvd – 接收的总 FCM 缓冲区数监视元素

buff_free – 当前可用的 FCM 缓冲区数

元素标识 buff_free

元素类型 标尺

表 183. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

描述 此元素指示当前可用的 FCM 缓冲区数。

用法 将当前可用的 FCM 缓冲区数与 *fcm_num_buffers* 配置参数配合使用来确定当前 FCM 缓冲池利用率。可使用此信息来调整 *fcm_num_buffers*。

相关参考:
- 第 210 页的『buff_free_bottom – 最少可用 FCM 缓冲区数』

buff_free_bottom – 最少可用 FCM 缓冲区数

元素标识 buff_free_bottom

元素类型 水位标记

表 184. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

描述 处理期间达到的最低可用 FCM 缓冲区数。

用法 将此元素与 *fcm_num_buffers* 配置参数一起使用来确定最大 FCM 缓冲池利用率。如果 buff_free_bottom 很低，则应提高 *fcm_num_buffers* 以确保操作不会用完 FCM 缓冲区。如果 buff_free_bottom 很高，可降低 *fcm_num_buffers* 以节约系统资源。

相关参考:
- 第 210 页的『buff_free – 当前可用的 FCM 缓冲区数』

ch_free － 当前可用的通道数

元素标识 ch_free

元素类型 标尺

表 185. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

描述 此元素指示当前可用的节点间通信信道数。

用法 将当前可用的通信信道数与 *fcm_num_channels* 配置参数一起使用来确定当前连接条目利用率。可使用此信息来调整 *fcm_num_channels*。

相关参考:

- 第 211 页的『ch_free_bottom － 最低可用通道数』
- 『fcm_num_channels － FCM 通道数配置参数』（*《性能指南》*）

ch_free_bottom － 最低可用通道数

元素标识 ch_free_bottom

元素类型 水位标记

表 186. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

描述 处理期间达到的最低可用节点间通信信道数。

用法 将此元素与 *fcm_num_channels* 配置参数一起使用来确定最大连接条目利用率。

相关参考:

- 第 211 页的『ch_free － 当前可用的通道数』
- 『fcm_num_channels － FCM 通道数配置参数』（*《性能指南》*）

connection_status － 连接状态

元素标识 connection_status

元素类型 信息

表 187. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本

描述 此元素指示发出 GET SNAPSHOT 命令的节点与 *db2nodes.cfg* 文件中列示的其他节点间的通信连接状态。

用法 连接值包括:

SQLM_FCM_CONNECT_INACTIVE
当前无连接

SQLM_FCM_CONNECT_ACTIVE
连接处于活动状态

SQLM_FCM_CONNECT_CONGESTED
连接已阻塞

两个节点可以处于活动状态，但除非节点之间存在通信，否则两个节点之间的通信连接仍然不活动。

相关参考:

- 第 212 页的『total_buffers_sent － 发送的总 FCM 缓冲区数』
- 第 212 页的『total_buffers_rcvd － 接收到的总 FCM 缓冲区数』

total_buffers_sent － 发送的总 FCM 缓冲区数

元素标识 total_buffers_sent

元素类型 计数器

表 188. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本

描述 从发出 GET SNAPSHOT 命令的节点发送至 *node_number*（请参阅 *db2nodes.cfg* 文件）标识的节点的总 FCM 缓冲区数。

用法 可使用此元素来量度当前节点与远程节点之间的通信量级别。如果发送至此节点的总 FCM 缓冲区数很高，您可能想要重新分发数据库或移动表以降低节点间通信量。

相关参考:

- 第 211 页的『connection_status － 连接状态』
- 第 212 页的『total_buffers_rcvd － 接收到的总 FCM 缓冲区数』

total_buffers_rcvd － 接收到的总 FCM 缓冲区数

元素标识 total_buffers_rcvd

元素类型 计数器

表 189. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本

描述 发出 GET SNAPSHOT 命令的节点从 *node_number*（请参阅 *db2nodes.cfg* 文件）标识的节点接收到的总 FCM 缓冲区数。

用法 可使用此元素来量度当前节点与远程节点之间的通信量级别。如果从此节点接收到的总 FCM 缓冲区数很高，您可能想要重新分发数据库或移动表以降低节点间通信量。

相关参考:

- 第 211 页的『connection_status － 连接状态』
- 第 212 页的『total_buffers_sent － 发送的总 FCM 缓冲区数』

实用程序

实用程序监视元素

下列元素提供有关实用程序的信息:
- utility_dbname – 实用程序操作的数据库监视元素utility_dbname – 实用程序操作的数据库监视元素
- utility_id – 实用程序标识监视元素utility_id – 实用程序标识监视元素
- utility_type – 实用程序类型监视元素utility_type – 实用程序类型监视元素
- utility_priority – 实用程序优先级监视元素utility_priority – 实用程序优先级监视元素
- utility_start_time – 实用程序启动时间监视元素utility_start_time – 实用程序启动时间监视元素
- utility_description – 实用程序描述监视元素utility_description – 实用程序描述监视元素
- progress_list_cur_seq_num – 当前进度列表序号监视元素progress_list_cur_seq_num – 当前进度列表序号监视元素
- progress_list_attr – 当前进度列表属性监视元素progress_list_attr – 当前进度列表属性监视元素
- progress_seq_num – 进度序号监视元素progress_seq_num – 进度序号监视元素
- progress_description – 进度描述监视元素progress_description – 进度描述监视元素
- progress_start_time – 进度开始时间监视元素progress_start_time – 进度开始时间监视元素
- progress_work_metric – 进度工作度量值监视元素progress_work_metric – 进度工作度量值监视元素
- progress_total_units – 总计进度工作单位监视元素progress_total_units – 总计进度工作单位监视元素
- progress_completed_units – 完成的进度工作单位监视元素progress_completed_units – 完成的进度工作单位监视元素

utility_dbname – 实用程序操作的数据库

元素标识 utility_dbname

元素类型 信息

表 190. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

描述 实用程序操作的数据库。

utility_id – 实用程序标识

元素标识 utility_id

元素类型 信息

表 191. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

描述 对应于实用程序调用的唯一标识。

utility_type － 实用程序类型

元素标识 utility_type

元素类型 信息

表 192. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

描述 实用程序的类。

用法

此元素的值（列示如下）是在 sqlmon.h 中定义的。

API 常量	实用程序
SQLM_UTILITY_REBALANCE	重新平衡
SQLM_UTILITY_BACKUP	备份
SQLM_UTILITY_RUNSTATS	Runstats
SQLM_UTILITY_REORG	重组
SQLM_UTILITY_RESTORE	复原
SQLM_UTILITY_CRASH_RECOVERY	崩溃恢复
SQLM_UTILITY_ROLLFORWARD_RECOVERY	前滚恢复
SQLM_UTILITY_LOAD	装入
SQLM_UTILITY_RESTART_RECREATE_INDEX	重新启动重新创建索引

utility_priority － 实用程序优先级

元素标识 utility_priority

元素类型 信息

表 193. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

描述 实用程序优先级指定调速实用程序相对其调速层而言的相对重要性。优先级 0 暗示实用程序以非调速方式执行。非零优先级的范围必须在 1 到 100 之间，100 表示最高优先级，1 表示最低优先级。

相关参考:

- 『LIST UTILITIES command』（*Command Reference*）
- 『SET UTIL_IMPACT_PRIORITY command』（*Command Reference*）
- 『util_impact_lim － 实例影响策略配置参数』（*《性能指南》*）

utility_start_time － 实用程序启动时间

元素标识 utility_start_time

元素类型 时间戳记

表 194. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

描述 初次调用当前实用程序的日期和时间。

utility_description – 实用程序描述

元素标识 utility_description

元素类型 信息

表 195. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

描述 对实用程序执行的任务的简短描述。例如，重新平衡调用可能包含"Tablespace ID: 2"，表示此重新平衡程序正在处理标识为 2 的表空间。此字段的格式取决于实用程序的类，并且在发行版之间可能更改。

progress_list_cur_seq_num – 当前进度列表序号

元素标识 progress_list_cur_seq_num

元素类型 信息

表 196. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress_list	基本

描述 如果实用程序包含多个顺序阶段，则此元素显示当前阶段的编号。

用法 使用此元素来确定多阶段实用程序的当前阶段。请参阅对 *progress_seq_num* 的描述。

相关参考:

• 第 216 页的『progress_seq_num – 进度序号』

progress_list_attr – 当前进度列表属性

元素标识 progress_list_attr

元素类型 信息

表 197. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress_list	基本

描述 此元素描述如何解释进度元素列表。此元素的值是下列其中一个常量:

- SQLM_ELM_PROGRESS_LIST_ATTR_SERIAL – 列表中的元素将解释为一组串行阶段，这意味着第一次更新元素 n+1 的已完成工作之前，已完成工作

必须等于元素 n 的总工作。此属性用于描述由一组串行阶段组成的任务的进度，其中串行阶段意味着必须彻底完成一个阶段才能开始下一个阶段。

- SQLM_ELM_PROGRESS_LIST_ATTR_CONCURRENT – 进度列表中的任何元素可以随时更新。

用法 使用此元素来确定更新 progress_list 的各个元素的方式。

相关参考:

- 第 215 页的『progress_list_cur_seq_num – 当前进度列表序号』

progress_seq_num – 进度序号

元素标识 progress_seq_num

元素类型 信息

表 198. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

描述 阶段号。

注: 仅对由多个执行阶段组成的实用程序显示阶段号。

用法 使用此元素来确定多阶段实用程序内的阶段顺序。该实用程序将按进度序号递增顺序来顺序执行各个阶段。可通过将 *progress_seq_num* 与 *progress_list_current_seq_num* 的值相匹配来找到多阶段实用程序的当前阶段。

相关参考:

- 第 215 页的『progress_list_cur_seq_num – 当前进度列表序号』

progress_description – 进度描述

元素标识 progress_description

元素类型 信息

表 199. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

描述 描述工作阶段。Load 实用程序的示例值包括:

- DELETE
- LOAD
- REDO

用法 使用此元素来获取对某个阶段的一般描述。

progress_start_time – 进度开始时间

元素标识 progress_start_time

元素类型 信息

表 200. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

描述 表示阶段开始的时间戳记。

用法 使用此元素来确定阶段开始的时间。如果该阶段尚未开始，则省略此元素。

progress_work_metric – 进度工作度量

元素标识 progress_work_metric

元素类型 信息

表 201. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

描述 解释 *progress_total_units* 和 *progress_completed_units* 元素的度量。示例值包括:

- SQLM_WORK_METRIC_BYTES
- SQLM_WORK_METRIC_EXTENTS

注:

1. 可能并非所有实用程序都包含此元素。
2. 此元素的值可在 sqlmon.h 中找到。

用法 使用此元素的值来确定用作报告度量的 *progress_total_units* 和 *progress_completed_units*。

相关参考:

- 第 218 页的『progress_completed_units – 完成的进度工作单元数』
- 第 217 页的『progress_total_units – 进度工作单元总数』

progress_total_units – 进度工作单元总数

元素标识 progress_total_units

元素类型 信息

表 202. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

描述 为了完成某个阶段而执行的工作总量。某些实用程序可能无法确定总工作量，所以它们将持续更新此元素。其他实用程序可能无法估计总工作量，所以可能完全省略此元素。

此元素在 *progress_work_metric* 监视元素中以单元表示。

用法 使用此元素来确定某个阶段的总工作量。将此元素与 *progress_completed_units* 配合使用来计算某个阶段完成的工作百分比:

```
percentage complete = progress_completed_units / progress_total_units* 100
```

相关参考:

- 第 218 页的『progress_completed_units – 完成的进度工作单元数』
- 第 217 页的『progress_work_metric – 进度工作度量』

progress_completed_units – 完成的进度工作单元数

元素标识	progress_completed_units
元素类型	信息

表 203. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

描述 当前阶段完成的工作单元数。此元素的值通常会在实用程序操作时递增。此元素总是小于或等于 *progress_total_units* （如果同时定义了这两个元素）。

注:

1. 可能并非所有实用程序都包含此元素。
2. 此元素在 *progress_work_metric* 监视元素中以单元表示。

用法 使用此元素来确定某个阶段完成的工作量。此元素本身可用来监视正在运行的实用程序的活动。实用程序执行时，此元素应不断递增。如果 *progress_completed_units* 在很长一段时间内未能递增，则实用程序可能已停止。

如果定义了 *progress_total_units*，则此元素可用来计算完成工作的百分比:

```
percentage complete = progress_completed_units / progress_total_units* 100
```

相关参考:

- 第 217 页的『progress_total_units – 进度工作单元总数』
- 第 217 页的『progress_work_metric – 进度工作度量』

utility_state – 实用程序状态

元素标识	utility_state
元素类型	信息

表 204. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

描述 此元素描述实用程序的状态。

用法 使用此元素来确定活动实用程序的状态。此字段的值（列示如下）是在 sqlmon.h 中定义的。

API 常量	描述
SQLM_UTILITY_STATE_EXECUTE	实用程序正在执行
SQLM_UTILITY_STATE_WAIT	实用程序正在等待事件发生，然后才会继续执行
SQLM_UTILITY_STATE_ERROR	实用程序遇到了错误

相关参考:

- 第 213 页的『utility_dbname － 实用程序操作的数据库』
- 第 215 页的『utility_description － 实用程序描述』
- 第 213 页的『utility_id － 实用程序标识』
- 第 214 页的『utility_type － 实用程序类型』

utility_invoker_type － 实用程序调用者类型

元素标识 utility_invoker_type

元素类型 信息

表 205. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

描述 此元素描述实用程序是如何被调用的。

用法 使用此元素来确定实用程序是如何被调用的。例如，可以使用此元素来确定实用程序是 DB2 自动调用的还是用户调用的。此元素的值（列示如下）是在 sqlmon.h 中定义的。

API 常量	实用程序
SQLM_UTILITY_INVOKER_USER	实用程序是用户调用的
SQLM_UTILITY_INVOKER_AUTO	实用程序是 DB2 自动调用的

相关参考:

- 第 213 页的『utility_dbname － 实用程序操作的数据库』
- 第 213 页的『utility_id － 实用程序标识』

数据库配置

数据库配置监视元素

下列元素提供对数据库性能调整特别有帮助的信息。

- 缓冲池活动监控器元素
- 非缓冲的 I/O 活动监控器元素
- 目录高速缓存监视元素
- 程序包高速缓存监视元素
- SQL 工作空间监视元素
- 数据库堆监视元素
- 记录监视元素

缓冲池活动

缓冲池活动监视元素

数据库服务器执行的所有数据读取和更新操作都是对缓冲池进行的。当应用程序需要数据时，就会将数据从磁盘复制到缓冲池。

下列程序将页放入缓冲池:

- 代理程序。这是同步 I/O。
- I/O 服务器（预取程序）。这是异步 I/O。

下列程序将页从缓冲池写入磁盘:

- 代理程序（同步方式）
- 页清除程序（异步方式）

如果服务器需要读取一页数据，并且该页已在缓冲池中，则访问该页的速度要比从磁盘读取该页快。如果在缓冲池中能**命中**尽可能多的页，那就最好不过了。避免磁盘 I/O 操作对于提高数据库性能来说十分重要，因此，在调整性能时，正确地配置缓冲池是其中一项最重要的考虑事项。

对于页请求，如果该页已在缓冲池中，数据库管理器就不需要从磁盘装入该页便可以为该请求提供服务，缓冲池命中率指的就是此类情况所占的百分比。缓冲池命中率越高，磁盘 I/O 操作频率就会越低。

缓冲池命中率的计算公式如下:

```
     1 - ((pool_data_p_reads + pool_xda_p_reads +
 pool_index_p_reads + pool_temp_data_p_reads
 + pool_temp_xda_p_reads + pool_temp_index_p_reads )
      / (pool_data_l_reads + pool_xda_l_reads + pool_index_l_reads +
      pool_temp_data_l_reads + pool_temp_xda_l_reads
      + pool_temp_index_l_reads )) * 100%
```

此计算公式考虑缓冲池高速缓存的所有页（索引和数据）。

另一种方便的方法是使用 BP_HITRATIO 管理视图来监视缓冲池命中率。

对于大型数据库来说，增加缓冲池大小对缓冲池命中率的影响极小。它的数据页数可能非常多，增加缓冲池大小并不会提高命中统计机率。而是，您会发现通过调整索引缓冲池命中率能获得期望的结果。这可以通过两种方法实现:

1. 将数据和索引分到两个不同的缓冲池中并分别对它们进行调整。
2. 使用一个缓冲池，但增加其大小，直到索引命中率不再提高为止。索引缓冲池命中率的计算公式如下:

   ```
   (1 - ((pool_index_p_reads) / (pool_index_l_reads))) * 100%
   ```

第一种方法通常效率更高，但由于它要求索引和数据在不同的表空间中，所以对于现有数据库来说不能选择此方法。此方法还要求调整两个缓冲池（而不是只调整一个缓冲池），此任务可能更难以完成，当内存有限时尤其如此。

您还应该考虑预取程序对命中率的影响。预取程序将数据页读入缓冲池，即预计应用程序需要这些数据页（异步方式）。在大多数情况下，这些页刚好是在需要它们之前读取的（期望的情况）。但是，预取程序会将不会使用到的页读入缓冲池，从而执行不必

要的 I/O 操作。例如，应用程序开始读整个表。这种情况被检测到，预取开始，但应用程序在填满应用程序缓冲区后停止读取。同时，预取操作已经读取了许多其他的页。已经为不会使用到的页执行了 I/O，那些页占用了缓冲池的部分空间。

页清除程序监视缓冲池并以异步方式将页写入磁盘。它们的目标是:

- 确保代理程序在缓冲池中总能找到可用页。如果代理程序在缓冲池中找不到可用页，它就必须自己清除它们，相关应用程序的响应速度就会变慢。
- 加快系统崩溃时的数据库恢复速度。已写入磁盘的页数越多，恢复数据库时必须处理的日志文件记录数就越少。

虽然会将脏页写入磁盘，但除非需要空间来读入新页，否则不会立即从缓冲池中除去这些页。

注: 缓冲池信息通常是在表空间级别收集的，但数据库系统监视器的工具可以提高到缓冲池级别和数据库级别来收集信息。根据分析类型的不同，可能需要在任何或全部这些级别检查此数据。

下列元素提供有关缓冲池活动的信息。

- bp_id – 缓冲池标识监视元素
- pool_data_l_reads – 逻辑读取的缓冲池数据页数监视元素
- pool_temp_data_l_reads – 逻辑读取的缓冲池临时数据页数监视元素 pool_temp_data_l_reads – 逻辑读取的缓冲池临时数据页数监视元素
- pool_data_p_reads – 物理读取的缓冲池数据页数监视元素
- pool_temp_data_p_reads – 物理读取的缓冲池临时数据页数监视元素 pool_temp_data_p_reads – 物理读取的缓冲池临时数据页数监视元素
- pool_data_writes – 缓冲池数据页写入数监视元素
- pool_index_l_reads – 逻辑读取的缓冲池索引页数监视元素
- pool_temp_index_l_reads – 逻辑读取的缓冲池临时索引页数监视元素 pool_temp_index_l_reads – 逻辑读取的缓冲池临时索引页数监视元素
- pool_index_p_reads – 物理读取的缓冲池索引页数监视元素
- pool_temp_index_p_reads – 物理读取的缓冲池临时索引页数监视元素 pool_temp_index_p_reads – 物理读取的缓冲池临时索引页数监视元素
- pool_index_writes – 写入缓冲池索引页的次数监视元素
- pool_xda_l_reads – 逻辑读取的缓冲池 XDA 数据页数监视元素pool_xda_l_reads – 逻辑读取的缓冲池 XDA 数据页数监视元素
- pool_temp_xda_l_reads – 逻辑读取的缓冲池临时 XDA 数据页数监视元素 pool_temp_xda_l_reads – 逻辑读取的缓冲池临时 XDA 数据页数监视元素
- pool_xda_p_reads – 物理读取的缓冲池 XDA 数据页数监视元素pool_xda_p_reads – 物理读取的缓冲池 XDA 数据页数监视元素
- pool_temp_xda_p_reads – 物理读取的缓冲池临时 XDA 数据页数监视元素 pool_temp_xda_p_reads – 物理读取的缓冲池临时 XDA 数据页数监视元素
- pool_xda_writes – 缓冲池 XDA 数据写入数监视元素pool_xda_writes – 缓冲池 XDA 数据写入数监视元素
- pool_read_time – 缓冲池总物理读取时间监视元素
- pool_write_time – 缓冲池总物理写入时间监视元素
- files_closed – 关闭的数据库文件数监视元素
- pool_async_data_reads – 缓冲池异步读取的数据页数监视元素
- pool_async_data_writes – 缓冲池异步写入的数据页数监视元素
- pool_async_index_writes – 缓冲池异步写入的索引页数监视元素

- pool_async_index_reads – 缓冲池异步读取的索引页数监视元素
- pool_async_xda_writes – 缓冲池异步 XDA 数据写入数监视元素pool_async_xda_writes – 缓冲池异步 XDA 数据写入数监视元素
- pool_async_xda_reads – 缓冲池异步 XDA 数据读取数监视元素pool_async_xda_reads – 缓冲池异步 XDA 数据读取数监视元素
- pool_async_read_time – 缓冲池异步读取时间监视元素
- pool_async_write_time – 缓冲池异步写入时间监视元素
- pool_async_data_read_reqs – 缓冲池异步读取请求数监视元素
- pool_async_index_read_reqs – 对索引页的缓冲池异步读取请求数监视元素 pool_async_index_read_reqs – 对索引页的缓冲池异步读取请求数监视元素
- pool_async_xda_read_reqs – 缓冲池异步 XDA 读取请求数监视元素 pool_async_xda_read_reqs – 缓冲池异步 XDA 读取请求数监视元素
- pool_lsn_gap_clns – 触发缓冲池日志空间清除程序的次数监视元素
- pool_drty_pg_steal_clns – 触发缓冲池干扰页清除程序的次数监视元素
- pool_no_victim_buffer – 缓冲池非牺牲性缓冲区监视元素pool_no_victim_buffer – 缓冲池非牺牲性缓冲区监视元素
- pool_drty_pg_thrsh_clns – 触发缓冲池阈值清除程序的次数监视元素
- bp_name – 缓冲池名称监视元素
- prefetch_wait_time – 等待预取的时间监视元素
- unread_prefetch_pages – 未读取的预取页数监视元素
- pages_from_vectored_ios – 向量 IO 读取的总页数监视元素
- block_ios – 块 IO 请求数监视元素
- vectored_ios – 向量 IO 请求数监视元素
- pages_from_block_ios – 块 IO 读取的总页数监视元素
- physical_page_maps – 物理页面映射数监视元素

相关概念:

- 『XML 存储器对象概述』(*《管理指南: 计划》*)

bp_id – 缓冲池标识

元素标识 bp_id

元素类型 信息

表 206. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	基本

描述 此元素包含正在监视的缓冲池的缓冲池标识。

pool_data_l_reads – 缓冲池数据页逻辑读取数

元素标识 pool_data_l_reads

元素类型 计数器

表 207. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池

表 207. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 208. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从缓冲池（逻辑）中对常规表空间和大型表空间请求的数据页的数目。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 此计数包括数据处于下列情况时对数据的访问:

- 当数据库管理器需要处理页时数据已经在缓冲池中
- 应读取到缓冲池中，数据库管理器才能处理页。

通过与 *pool_data_p_reads* 一起使用，可以借助以下公式来计算缓冲池的数据页命中率:

```
1 - (pool_data_p_reads / pool_data_l_reads)
```

有关确定整体缓冲池命中率的信息，请参阅*缓冲池活动监视元素*。

增加缓冲池大小一般会改进命中率，但您会达到一个最优状态，而无法继续改进。从理论上说，如果能够分配大到足以存储整个数据库的缓冲池，则系统启动并运行后你可以得到 100% 的命中率。但在许多情况下这是不现实的。命中率的高低实际上取决于数据的大小以及访问数据的方式。如果数据库很大并且数据访问比较平均，则命中率将会很低。对于非常大的表来说，您几乎无能为力。在此情况下，应该将重点放在较小并且访问较为频繁的表以及索引上。如果可能的话，将它们指定给希望获取高命中率的各个缓冲池。

相关参考:

- 第 224 页的『pool_data_p_reads – 缓冲池数据页物理读取数』
- 第 226 页的『pool_data_writes – 缓冲池数据页写入数』
- 第 227 页的『pool_index_l_reads – 缓冲池索引逻辑读取数』
- 第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』
- 第 224 页的『pool_temp_data_l_reads – 缓冲池临时数据逻辑读取数』
- 第 176 页的『appl_con_time – 连接请求启动时间戳记』
- 第 150 页的『db_conn_time – 数据库激活时间戳记』
- 第 220 页的『缓冲池活动监视元素』

pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数

元素标识 pool_temp_data_l_reads

元素类型 计数器

表 209. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 210. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从缓冲池（逻辑）中对临时表空间请求的数据页的数目。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法

与 *pool_temp_data_p_reads* 元素配合使用，可使用以下公式对临时表空间中的缓冲池计算数据页命中率：

```
1 - (pool_temp_data_p_reads / pool_temp_data_l_reads)
```

有关确定整体缓冲池命中率的信息，请参阅*缓冲池活动监视元素*。

相关参考:

- 第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』
- 第 225 页的『pool_temp_data_p_reads － 缓冲池临时数据物理读取数』
- 第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』
- 第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』
- 第 220 页的『缓冲池活动监视元素』

pool_data_p_reads － 缓冲池数据页物理读取数

元素标识 pool_data_p_reads

元素类型 计数器

表 211. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 212. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从表空间容器（物理）中对常规表空间和大型表空间读取的数据页的数目。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 有关如何使用此元素的信息，请参阅 pool_data_l_reads 和 pool_async_data_reads。

相关参考:

- 第 234 页的『pool_async_data_reads – 缓冲池异步数据读取数』
- 第 222 页的『pool_data_l_reads – 缓冲池数据页逻辑读取数』
- 第 226 页的『pool_data_writes – 缓冲池数据页写入数』
- 第 227 页的『pool_index_l_reads – 缓冲池索引逻辑读取数』
- 第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』
- 第 225 页的『pool_temp_data_p_reads – 缓冲池临时数据物理读取数』
- 第 176 页的『appl_con_time – 连接请求启动时间戳记』
- 第 150 页的『db_conn_time – 数据库激活时间戳记』

pool_temp_data_p_reads – 缓冲池临时数据物理读取数

元素标识 pool_temp_data_p_reads

元素类型 计数器

表 213. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

表 213. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 214. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从表空间容器（物理）中对临时表空间读取的数据页的数目。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 有关如何使用此元素的信息，请参阅 *pool_temp_data_l_reads*。

相关参考:

- 第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』
- 第 224 页的『pool_temp_data_l_reads － 缓冲池临时数据逻辑读取数』
- 第 228 页的『pool_temp_index_l_reads － 缓冲池临时索引逻辑读取数』
- 第 230 页的『pool_temp_index_p_reads － 缓冲池临时索引物理读取数』

pool_data_writes － 缓冲池数据页写入数

元素标识 pool_data_writes

元素类型 计数器

表 215. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 216. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

描述 指示缓冲池数据页物理写至磁盘的次数。

用法 如果缓冲池数据页写入磁盘在 pool_data_p_reads 中占很高的百分比，可通过提高可用于数据库的缓冲池页数来改进性能。

缓冲池数据页将因为下列原因写至磁盘:
- 释放缓冲池中的页以便可读取另一页
- 清空缓冲池。

系统不会总是写入页以便为新页腾出空间。如果页未更新，只要替换它就可以了。此元素不考虑此替换。

在需要缓冲池空间之前，数据页可由异步页清除程序代理程序写入。这些异步页写入与同步页写入一起包括在此元素的值中（请参阅 pool_async_data_writes）。

计算此百分比时，忽略一开始填充缓冲池所需的物理读取的数目。要确定写入的页数:
1. 运行应用程序（以装入缓冲区）
2. 注意此元素的值
3. 再次运行应用程序
4. 从此元素的新值中减去步骤 2 中记录的值。

为避免在应用程序的各次运行之间释放缓冲池，应该采取下列其中一个操作:
- 使用 ACTIVATE DATABASE 命令激活数据库
- 将空闲应用程序连接至数据库。

如果所有应用程序都要更新该数据库，则提高缓冲池大小对性能可能没有多大影响，原因是大多数缓冲池页包含已更新数据，而这些数据必须写入磁盘。但是，如果已更新页在写出之前可供其他工作单元使用，则缓冲池可保存读写操作，这将对性能有所改进。

有关缓冲池大小的更多信息，请参阅*管理指南*。

相关参考:

- 第 150 页的『db_conn_time － 数据库激活时间戳记』
- 第 176 页的『appl_con_time － 连接请求启动时间戳记』
- 第 222 页的『pool_data_l_reads － 缓冲池数据页逻辑读取数』
- 第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』
- 第 233 页的『pool_write_time － 缓冲池物理写入总时间』
- 第 235 页的『pool_async_data_writes － 缓冲池异步数据写入数』

pool_index_l_reads － 缓冲池索引逻辑读取数

元素标识 pool_index_l_reads

元素类型 计数器

表 217. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池

表 217. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 218. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从缓冲池（逻辑）中对常规表空间和大型表空间请求的索引页的数目。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 此计数包括索引页处于下列情况时对索引页的访问:
- 当数据库管理器需要处理页时索引页已经在缓冲池中
- 应读取到缓冲池中，数据库管理器才能处理页。

通过与 pool_index_p_reads 一起使用，可以借助下列其中一项来计算缓冲池的索引页命中率:

```
1 - (pool_index_p_reads / pool_index_l_reads)
```

要计算整体缓冲池命中率，请参阅 pool_data_l_reads。

如果命中率很低，则提高缓冲池页数可以改进性能。有关缓冲池大小的更多信息，请参阅*管理指南*。

相关参考:

- 第 222 页的『pool_data_l_reads – 缓冲池数据页逻辑读取数』
- 第 224 页的『pool_data_p_reads – 缓冲池数据页物理读取数』
- 第 226 页的『pool_data_writes – 缓冲池数据页写入数』
- 第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』
- 第 231 页的『pool_index_writes – 缓冲池索引写入数』
- 第 230 页的『pool_temp_index_p_reads – 缓冲池临时索引物理读取数』
- 第 176 页的『appl_con_time – 连接请求启动时间戳记』
- 第 150 页的『db_conn_time – 数据库激活时间戳记』

pool_temp_index_l_reads – 缓冲池临时索引逻辑读取数

元素标识 pool_temp_index_l_reads

元素类型 计数器

表 219. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

表 219. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 220. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从缓冲池（逻辑）中对临时表空间请求的索引页的数目。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 有关如何使用此元素的信息，请参阅 *pool_temp_data_l_reads*。

相关参考:

- 第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』
- 第 224 页的『pool_temp_data_l_reads – 缓冲池临时数据逻辑读取数』
- 第 225 页的『pool_temp_data_p_reads – 缓冲池临时数据物理读取数』
- 第 230 页的『pool_temp_index_p_reads – 缓冲池临时索引物理读取数』

pool_index_p_reads – 缓冲池索引物理读取数

元素标识 pool_index_p_reads

元素类型 计数器

表 221. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 222. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

表 222. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从表空间容器（物理）中对常规表空间和大型表空间读取的索引页的数目。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 有关如何使用此元素的信息，请参阅 pool_index_l_reads。

相关参考:

- 第 222 页的『pool_data_l_reads – 缓冲池数据页逻辑读取数』
- 第 224 页的『pool_data_p_reads – 缓冲池数据页物理读取数』
- 第 227 页的『pool_index_l_reads – 缓冲池索引逻辑读取数』
- 第 231 页的『pool_index_writes – 缓冲池索引写入数』
- 第 228 页的『pool_temp_index_l_reads – 缓冲池临时索引逻辑读取数』
- 第 176 页的『appl_con_time – 连接请求启动时间戳记』
- 第 150 页的『db_conn_time – 数据库激活时间戳记』

pool_temp_index_p_reads – 缓冲池临时索引物理读取数

元素标识 pool_temp_index_p_reads

元素类型 计数器

表 223. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 224. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从表空间容器（物理）中对临时表空间读取的索引页的数目。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 有关如何使用此元素的信息，请参阅 *pool_temp_data_l_reads*。

相关参考:

- 第 227 页的『pool_index_l_reads – 缓冲池索引逻辑读取数』
- 第 224 页的『pool_temp_data_l_reads – 缓冲池临时数据逻辑读取数』
- 第 225 页的『pool_temp_data_p_reads – 缓冲池临时数据物理读取数』
- 第 228 页的『pool_temp_index_l_reads – 缓冲池临时索引逻辑读取数』

pool_index_writes – 缓冲池索引写入数

元素标识 pool_index_writes

元素类型 计数器

表 225. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 226. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

描述 指示缓冲池索引页物理写至磁盘的次数。

用法 与数据页一样，缓冲池索引页将因为下列原因写至磁盘:
- 释放缓冲池中的页以便可读取另一页
- 清空缓冲池。

系统不会总是写入页以便为新页腾出空间。如果页未更新，只要替换它就可以了。此元素不考虑此替换。

在需要缓冲池空间之前，索引页可由异步页清除程序代理程序写入。这些异步索引页写入与同步索引页写入一起包括在此元素的值中（请参阅 pool_async_index_writes）。

如果缓冲池索引页写入磁盘在 *pool_index_p_reads* 中占很高的百分比，可通过提高可用于数据库的缓冲池页数来改进性能。

计算此百分比时，忽略一开始填充缓冲池所需的物理读取的数目。要确定写入的页数:
1. 运行应用程序（以装入缓冲区）
2. 注意此元素的值
3. 再次运行应用程序
4. 从此元素的新值中减去步骤 2 中记录的值。

为避免在应用程序的各次运行之间释放缓冲池，应该采取下列其中一个操作:

- 使用 ACTIVATE DATABASE 命令激活数据库
- 将空闲应用程序连接至数据库。

如果所有应用程序都要更新该数据库，则提高缓冲池大小对性能可能没有多大影响，原因是大多数页包含已更新数据，而这些数据必须写入磁盘。

有关缓冲池大小的更多信息，请参阅*管理指南*。

相关参考:

- 第 150 页的『db_conn_time – 数据库激活时间戳记』
- 第 176 页的『appl_con_time – 连接请求启动时间戳记』
- 第 227 页的『pool_index_l_reads – 缓冲池索引逻辑读取数』
- 第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』
- 第 235 页的『pool_async_index_writes – 缓冲池异步索引写入数』

pool_read_time – 缓冲池物理读总时间

元素标识 pool_read_time

元素类型 计数器

表 227. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 228. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

描述 指示从所有类型的表空间的表空间容器（物理容器）读取数据和索引页时耗费的总时间。此值以毫秒计。

用法 可以将此元素与 *pool_data_p_reads* 和 *pool_index_p_reads* 配合使用以计算平均页读取时间。此平均值非常重要，它表示存在 I/O 等待状态，而 I/O 等待状态又表示应该将数据移至另一设备。

在数据库和表空间级别，此元素包括 *pool_async_read_time* 的值。

相关参考:

- 第 224 页的『pool_data_p_reads – 缓冲池数据页物理读取数』
- 第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』
- 第 150 页的『db_conn_time – 数据库激活时间戳记』
- 第 176 页的『appl_con_time – 连接请求启动时间戳记』

- 第 237 页的『pool_async_read_time － 缓冲池异步读取时间』

pool_write_time － 缓冲池物理写入总时间

元素标识 pool_write_time

元素类型 计数器

表 229. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 230. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

描述 提供以物理方式将缓冲池中的数据或索引页写至磁盘时耗用的总时间。耗用时间以毫秒计。

用法 可以将此元素与 *buffer_pool_data_writes* 和 *pool_index_writes* 配合使用以计算平均页写时间。此平均值非常重要，它表示存在 I/O 等待状态，而 I/O 等待状态又表示应该将数据移至另一设备。

在数据库和表空间级别，此元素包括 *pool_async_write_time* 的值。

相关参考:

- 第 226 页的『pool_data_writes － 缓冲池数据页写入数』
- 第 231 页的『pool_index_writes － 缓冲池索引写入数』
- 第 150 页的『db_conn_time － 数据库激活时间戳记』
- 第 176 页的『appl_con_time － 连接请求启动时间戳记』

files_closed － 关闭的数据库文件数

元素标识 files_closed

元素类型 计数器

表 231. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 232. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 关闭的总数据库文件数。

用法 数据库管理器打开文件以便读入缓冲池和写出缓冲池。任何时间应用程序打开的最大数据库文件数都由 *maxfilop* 配置参数控制。如果达到最大文件数，则在打开新文件之前必须先关闭某个文件。注意，实际的打开文件数可能不等于关闭文件数。

可使用此元素来帮助您确定 *maxfilop* 配置参数的最佳值（有关更多信息，请参阅*管理指南*）。

pool_async_data_reads － 缓冲池异步数据读取数

元素标识 pool_async_data_reads

元素类型 计数器

表 233. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 234. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 指示从表空间容器（物理）中由异步引擎可派遣单元（EDU）对所有类型的表空间读取的数据页的数目。

用法 可将此元素与 pool_data_p_reads 配合使用以计算同步执行的物理读取数（即数据库管理器代理程序执行的物理数据页读取数）。使用以下公式:

```
pool_data_p_reads + pool_temp_data_p_reads - pool_async_data_reads
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 *num_ioservers* 配置参数（请参阅*管理指南*）时，此元素会很有帮助。

异步读取是由数据库管理器预取程序执行的。有关这些预取程序的信息，请参阅*管理指南*。

相关参考:

- 第 237 页的『pool_async_read_time － 缓冲池异步读取时间』
- 第 224 页的『pool_data_p_reads － 缓冲池数据页物理读取数』

- 第 239 页的『pool_async_data_read_reqs – 缓冲池异步读取请求数』
- 第 255 页的『direct_reads – 直接从数据库进行读取的读取操作数』

pool_async_data_writes – 缓冲池异步数据写入数

元素标识 pool_async_data_writes

元素类型 计数器

表 235. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 236. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 异步页清除程序或预取程序将缓冲池数据页物理写至磁盘的次数。预取程序可将脏页写至磁盘以便为要预取的页腾出空间。

用法 可将此元素与 buffer_pool_data_writes 配合使用以计算同步执行的物理写入请求数（即由数据库管理器代理程序执行的物理数据页写入数）。使用以下公式:

```
pool_data_writes - pool_async_data_writes
```

通过比较异步写入数与同步写入数的比率，可了解缓冲池页清除程序的执行情况。在调整 *num_iocleaners* 配置参数时，此比率会非常有用。

有关异步页清除程序的更多信息，请参阅*管理指南*。

相关参考:

- 第 235 页的『pool_async_index_writes – 缓冲池异步索引写入数』
- 第 226 页的『pool_data_writes – 缓冲池数据页写入数』
- 第 238 页的『pool_async_write_time – 缓冲池异步写入时间』
- 第 240 页的『pool_lsn_gap_clns – 触发的缓冲池日志空间清除程序数』
- 第 240 页的『pool_drty_pg_steal_clns – 触发的缓冲池牺牲页清除程序数』
- 第 242 页的『pool_drty_pg_thrsh_clns – 触发的缓冲池阈值清除程序数』
- 第 256 页的『direct_writes – 直接写入数据库的写入操作数』

pool_async_index_writes – 缓冲池异步索引写入数

元素标识 pool_async_index_writes

元素类型 计数器

表 237. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 238. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 异步页清除程序或预取程序将缓冲池索引页物理写至磁盘的次数。预取程序可将脏页写至磁盘以便为要预取的页腾出空间。

用法 可将此元素与 pool_index_writes 配合使用以计算同步执行的物理索引写入请求数。即，数据库管理器代理程序执行的物理索引页写入数。使用以下公式:

```
pool_index_writes - pool_async_index_writes
```

通过比较异步写入数与同步写入数的比率，可了解缓冲池页清除程序的执行情况。在调整 *num_iocleaners* 配置参数时，此比率会非常有用。

有关异步页清除程序的更多信息，请参阅*管理指南*。

相关参考:

- 第 235 页的『pool_async_data_writes － 缓冲池异步数据写入数』
- 第 236 页的『pool_async_index_reads － 缓冲池异步索引读取数』
- 第 231 页的『pool_index_writes － 缓冲池索引写入数』
- 第 238 页的『pool_async_write_time － 缓冲池异步写入时间』
- 第 240 页的『pool_lsn_gap_clns － 触发的缓冲池日志空间清除程序数』
- 第 240 页的『pool_drty_pg_steal_clns － 触发的缓冲池牺牲页清除程序数』
- 第 242 页的『pool_drty_pg_thrsh_clns － 触发的缓冲池阈值清除程序数』
- 第 256 页的『direct_writes － 直接写入数据库的写入操作数』

pool_async_index_reads － 缓冲池异步索引读取数

元素标识 pool_async_index_reads

元素类型 计数器

表 239. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 240. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 指示从表空间容器（物理）中由异步引擎可派遣单元（EDU）对所有类型的表空间读取的索引页的数目。

用法 可将此元素与 pool_index_p_reads 配合使用以计算同步执行的物理读取数（即数据库管理器代理程序执行的物理索引页读取数）。使用以下公式:

```
pool_index_p_reads + pool_temp_index_p_reads - pool_async_index_reads
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 *num_ioservers* 配置参数（请参阅*管理指南*）时，此元素会很有帮助。

异步读取是由数据库管理器预取程序执行的。有关这些预取程序的信息，请参阅*管理指南*。

相关参考:

- 第 235 页的『pool_async_data_writes – 缓冲池异步数据写入数』
- 第 235 页的『pool_async_index_writes – 缓冲池异步索引写入数』
- 第 229 页的『pool_index_p_reads – 缓冲池索引物理读取数』
- 第 237 页的『pool_async_read_time – 缓冲池异步读取时间』
- 第 240 页的『pool_lsn_gap_clns – 触发的缓冲池日志空间清除程序数』
- 第 240 页的『pool_drty_pg_steal_clns – 触发的缓冲池牺牲页清除程序数』
- 第 242 页的『pool_drty_pg_thrsh_clns – 触发的缓冲池阈值清除程序数』
- 第 255 页的『direct_reads – 直接从数据库进行读取的读取操作数』

pool_async_read_time – 缓冲池异步读取时间

元素标识	pool_async_read_time
元素类型	计数器

表 241. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 242. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 指示从表空间容器（物理）中由异步引擎可派遣单元（EDU）对所有类型的表空间读取数据和索引页所花费的总时间。此值以毫秒计。

用法 可使用此元素并借助以下公式来计算同步读取所耗用的时间:

```
pool_read_time - pool_async_read_time
```

还可使用此元素并借助以下公式来计算平均异步读取时间:

```
pool_async_read_time / pool_async_data_reads
```

这些计算可用来了解要执行的 I/O 处理。

相关参考:

- 第 234 页的『pool_async_data_reads – 缓冲池异步数据读取数』
- 第 232 页的『pool_read_time – 缓冲池物理读总时间』
- 第 239 页的『pool_async_data_read_reqs – 缓冲池异步读取请求数』
- 第 258 页的『direct_read_time – 直接读取时间』

pool_async_write_time – 缓冲池异步写入时间

元素标识	pool_async_write_time
元素类型	计数器

表 243. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 244. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 数据库管理器页清除程序将数据或索引页从缓冲池写至磁盘的总耗用时间。

用法 要计算同步写入页所耗用的时间，请使用以下公式:

```
pool_write_time - pool_async_write_time
```

还可使用此元素并借助以下公式来计算平均异步读取时间:

```
pool_async_write_time
/ (pool_async_data_writes
  + pool_async_index_writes)
```

这些计算可用来了解要执行的 I/O 处理。

相关参考:

- 第 235 页的『pool_async_data_writes – 缓冲池异步数据写入数』
- 第 235 页的『pool_async_index_writes – 缓冲池异步索引写入数』

- 第 233 页的『pool_write_time － 缓冲池物理写入总时间』
- 第 239 页的『pool_async_data_read_reqs － 缓冲池异步读取请求数』
- 第 259 页的『direct_write_time － 直接写入时间』

pool_async_data_read_reqs － 缓冲池异步读取请求数

元素标识 pool_async_data_read_reqs

元素类型 计数器

表 245. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 246. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 异步读取请求数。

用法 要计算每个异步请求读取的平均数据页数，请使用以下公式:

```
pool_async_data_reads / pool_async_data_read_reqs
```

此平均数可帮助您确定每次与预取程序交互时完成的异步 I/O 量。

相关参考:

- 第 234 页的『pool_async_data_reads － 缓冲池异步数据读取数』

pool_async_index_read_reqs － 缓冲池异步索引读取请求数

元素标识 pool_async_index_read_reqs

元素类型 计数器

表 247. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 248. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 针对索引页的异步读取请求的数目。

用法 要计算每个异步请求读取的索引页数，请使用以下公式:

```
pool_async_index_reads / pool_async_index_read_reqs
```

此平均数可帮助您确定每次与预取程序交互时对索引页进行的异步 I/O 量。

相关参考:

- 第 236 页的『pool_async_index_reads – 缓冲池异步索引读取数』

pool_lsn_gap_clns – 触发的缓冲池日志空间清除程序数

元素标识 pool_lsn_gap_clns

元素类型 计数器

表 249. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

可将快照监视的计数器复位。

表 250. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 因为使用的记录空间已达到数据库的预定义条件而调用页清除程序的次数。

用法 此元素可用来帮助评估您是否具有足够的记录空间以及您是否需要更多或更大的日志文件。

页清除条件将由 *softmax* 配置参数的设置确定。如果缓冲池中最旧的页包含一个更新，此更新由比条件值定义的当前日志位置还要旧的日志记录描述，则将触发页清除程序。有关更多信息，请参阅*管理指南*。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时:

- *pool_lsn_gap_clns* 监视元素将插入到监视器流中。
- 如果缓冲池中最旧的页包含一个更新，此更新由比条件值定义的当前日志位置还要旧的日志记录描述，则将触发页清除程序。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时:

- *pool_lsn_gap_clns* 监视元素会将 0 插入到监视器流中。
- 页清除程序主动写入页而不是等待条件值触发。

相关参考:

- 第 240 页的『pool_drty_pg_steal_clns – 触发的缓冲池牺牲页清除程序数』
- 第 242 页的『pool_drty_pg_thrsh_clns – 触发的缓冲池阈值清除程序数』
- 『性能变量』（*《性能指南》*）
- 『chngpgs_thresh – 已更改页数的阈值配置参数』（*《性能指南》*）

pool_drty_pg_steal_clns – 触发的缓冲池牺牲页清除程序数

元素标识 pool_drty_pg_steal_clns

元素类型 计数器

表 251. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

可将快照监视的计数器复位。

表 252. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 因为数据库的牺牲缓冲区替换期间需要同步写入而调用页清除程序的次数。

用法 通过使用以下公式，可以计算通过此元素表示的所有清除程序调用的百分比:

```
   pool_drty_pg_steal_clns
/ (pool_drty_pg_steal_clns
+  pool_drty_pg_thrsh_clns
+  pool_lsn_gap_clns)
```

如果此比率很低，则可能指示您定义的页清除程序过多。如果 *chngpgs_thresh* 设置得过低，则可能会写出将使其成为脏页的页。主动清除使得缓冲池失去了尽可能延迟写入这一用途。

如果此比率很高，则可能指示您定义的页清除程序过少。如果页清除程序过少，则故障后的恢复时间会增加（请参阅*管理指南*）。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时:

- *pool_drty_pg_steal_clns* 监视元素将插入到监视器流中。
- *pool_drty_pg_steal_clns* 监视元素计算因为数据库的牺牲缓冲区替换期间需要同步写入而调用页清除程序的次数。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时:

- *pool_drty_pg_steal_clns* 监视元素将 0 插入到监视器流中。
- 牺牲缓冲区替换期间需要同步写入时不会显式触发页清除程序。为确定对数据库或特定缓冲池配置的页清除程序数是否正确，请参阅 pool_no_victim_buffer 监视元素。

注: 虽然会将脏页写入磁盘，但除非需要空间来读入新页，否则不会立即从缓冲池中除去这些页。

相关参考:

- 『chngpgs_thresh － 已更改页数的阈值配置参数』（《*性能指南*》）
- 『性能变量』（《*性能指南*》）
- 第 242 页的『pool_drty_pg_thrsh_clns － 触发的缓冲池阈值清除程序数』
- 第 240 页的『pool_lsn_gap_clns － 触发的缓冲池日志空间清除程序数』
- 第 241 页的『pool_no_victim_buffer － 缓冲池没有牺牲缓冲区的次数』

pool_no_victim_buffer － 缓冲池没有牺牲缓冲区的次数

元素标识 pool_no_victim_buffer

元素类型 计数器

表 253. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 254. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 代理程序没有预先选择的可用牺牲缓冲区的次数。

用法 此元素可用来帮助评估使用前摄页清除时您是否具有足够的页清除程序用于给定缓冲池。

DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时，pool_no_victim_buffer 元素计算代理程序找不到预先选择的牺牲缓冲区可供立即使用，并且强制搜索缓冲池以查找适合的牺牲缓冲区的次数。

如果 pool_no_victim_buffer 元素的值相对于缓冲池中的逻辑读取数过高，则 DB2 数据库系统难以确保有足够的良好牺牲缓冲区可供使用。增加页清除程序数目将增加 DB2 提供预选牺牲缓冲区的能力。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时，pool_no_victim_buffer 元素没有预测值，并且可以安全地忽略它。在此配置中，DB2 数据库系统不会尝试确保代理程序预选可供使用的牺牲缓冲区，所以对缓冲池的大多数访问需要代理程序搜索缓冲池以查找牺牲缓冲区。

相关参考:

- 第 240 页的『pool_drty_pg_steal_clns – 触发的缓冲池牺牲页清除程序数』
- 第 242 页的『pool_drty_pg_thrsh_clns – 触发的缓冲池阈值清除程序数』
- 第 240 页的『pool_lsn_gap_clns – 触发的缓冲池日志空间清除程序数』

pool_drty_pg_thrsh_clns – 触发的缓冲池阈值清除程序数

元素标识 pool_drty_pg_thrsh_clns

元素类型 计数器

表 255. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

可将快照监视的计数器复位。

表 256. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 因为缓冲池达到数据库的脏页阈值条件而调用页清除程序的次数。

用法 该阈值由 *chngpgs_thresh* 配置参数设置。它是应用于缓冲池大小的百分比。如果池中的脏页数超过此值，将触发清除程序。

如果此值设置得过低，则页可能会过早写出，从而需要重新读入它们。如果此值设置得过高，则可能有太多的页累积，从而需要用户将这些页同步写出。有关更多信息，请参阅*管理指南*。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时:

- *pool_drty_pg_thrsh_clns* 监视元素将插入到监视器流中。
- *pool_drty_pg_thrsh_clns* 监视元素计算因为缓冲池达到数据库的脏页阈值条件而调用页清除程序的次数。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时:

- *pool_drty_pg_thrsh_clns* 监视元素将 0 插入到监视器流中。
- 页清除程序总是处于活动状态，以试图确保有足够的可用缓冲区以供牺牲，而不是等待条件值触发。

相关参考:

- 第 240 页的『pool_lsn_gap_clns – 触发的缓冲池日志空间清除程序数』
- 『chngpgs_thresh – 已更改页数的阈值配置参数』（*《性能指南》*）
- 『性能变量』（*《性能指南》*）

bp_name – 缓冲池名称

元素标识 bp_name

元素类型 信息

表 257. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	基本

描述 缓冲池的名称。

用法 每个数据库都至少需要一个缓冲池。根据您的需要，可以选择对单个数据库创建若干个大小不同的缓冲池。CREATE、ALTER 和 DROP BUFFERPOOL 语句允许您创建、更改或除去缓冲池。

创建新数据库后，它将具有缺省缓冲池 IBMDEFAULTBP，其大小将由平台确定。它还会具有一组系统缓冲池，每个系统缓冲池对应不同页大小:

- IBMSYSTEMBP4K
- IBMSYSTEMBP8K
- IBMSYSTEMBP16K
- IBMSYSTEMBP32K

不能改变这些系统缓冲池。

prefetch_wait_time – 等待预取的时间

元素标识 prefetch_wait_time

元素类型 计数器

表 258. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
应用程序	appl	缓冲池

表 259. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 应用程序等待 I/O 服务器（预取程序）完成将页装入到缓冲池中的操作所花的时间。

用法 可通过更改 I/O 服务器数目和 I/O 服务器大小来使用此元素进行试验。

unread_prefetch_pages – 未读取的预取页数

元素标识 unread_prefetch_pages

元素类型 计数器

表 260. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 261. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

描述 指示预取读入但从未使用的页数。

用法 如果此数目很高，则预取程序会将不会使用到的页读入缓冲池，从而导致执行不必要的 I/O。有关预取的更多信息，请参阅*管理指南*。

vectored_ios – 向量 IO 请求数

元素标识 vectored_ios

元素类型 标尺

表 262. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

描述 向量 I/O 请求的数目。更具体地说，就是 DB2 在缓冲池的页区域中执行顺序页预取的次数。

用法 使用此元素来确定执行向量 I/O 的频率。仅在顺序预取期间才监视向量 I/O 请求的数目。

相关参考:

- 第 245 页的『block_ios – 块 IO 请求数』
- 第 246 页的『pages_from_block_ios – 块 IO 读取的总页数』
- 第 245 页的『pages_from_vectored_ios – 向量 IO 读取的总页数』

pages_from_vectored_ios – 向量 IO 读取的总页数

元素标识	pages_from_vectored_ios
元素类型	标尺

表 263. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

描述 向量 I/O 读取到缓冲池的页区域中的总页数。

相关参考:

- 第 245 页的『block_ios – 块 IO 请求数』
- 第 244 页的『vectored_ios – 向量 IO 请求数』
- 第 246 页的『pages_from_block_ios – 块 IO 读取的总页数』

block_ios – 块 IO 请求数

元素标识	block_ios
元素类型	计数器

表 264. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

描述 块 I/O 请求的数目。更具体地说，就是 DB2 在缓冲池的块区域中执行顺序页预取的次数。

用法 如果启用了基于块的缓冲池，则此监视元素将报告执行块 I/O 的频率。否则，此监视元素将返回 0。在使用基于块的缓冲池时，只有在顺序预取期间才监视块 I/O 请求的数目。

如果启用了基于块的缓冲池，并且此数目很低或者接近向量 I/O 的数目（向量 IO 请求数监视元素的值），则考虑更改块大小。此状态可能指示下列其中一项:

- 绑定至缓冲池的一个或多个表空间的扩展数据块大小小于对缓冲池指定的块大小。
- 预取请求中请求的某些页已存在于缓冲池的页区域中。

预取程度允许在每个缓冲池块中浪费一些页，但如果浪费的页数过多，则预取程序将决定在缓冲池的页区域中执行向量 I/O。

为了更好地利用基于块的缓冲池提供的顺序预取性能改进，应对块大小选择适当的值。但是，因为带有不同扩展数据块大小的多个表空间可能绑定至同一个基于块的缓冲池，所以这一点可能比较难以做到。为了获取最佳性能，建议将具有相同扩展数据块大小的表空间绑定至一个基于块的缓冲池，该缓冲池的块大小等于扩展数据块大小。如果表空间的扩展数据块大小大于块大小，则可以获得较好的性能，扩展数据块大小小于块大小时情况则相反。

例如，如果扩展数据块大小为 2 而块大小为 8，则将使用向量 I/O 而不是块 I/O（块 I/O 会浪费 6 页）。将块大小降低至 2 将解决此问题。

相关参考:

- 第 244 页的『vectored_ios － 向量 IO 请求数』
- 第 246 页的『pages_from_block_ios － 块 IO 读取的总页数』
- 第 245 页的『pages_from_vectored_ios － 向量 IO 读取的总页数』

pages_from_block_ios － 块 IO 读取的总页数

元素标识 pages_from_block_ios

元素类型 计数器

表 265. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

描述 块 I/O 读取到缓冲池的块区域中的总页数。

用法 如果启用了基于块的缓冲池，则此元素将包含块 I/O 读取的总页数。否则此元素将返回 0。

pages_from_block_ios 除以 *block_ios* 元素将给出对每个基于块的 I/O 顺序预取的平均页数。如果 *pages_from_block_ios* 除以 *block_ios* 比对基于块的缓冲池定义的 BLOCKSIZE 小很多，则表示没有很好地利用基于块的 I/O 来。出现这种情况的一个可能原因是要顺序预取的表空间的扩展数据块大小与基于块的缓冲池的块大小之间不匹配。

相关参考:

- 第 245 页的『block_ios － 块 IO 请求数』
- 第 244 页的『vectored_ios － 向量 IO 请求数』
- 第 245 页的『pages_from_vectored_ios － 向量 IO 读取的总页数』

physical_page_maps － 物理页映射数

元素标识 physical_page_maps

元素类型 标尺

表 266. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

描述　物理页映射数。

pool_async_xda_read_reqs － 缓冲池异步 XDA 读取请求数

元素标识　pool_async_xda_read_reqs

元素类型　计数器

表 267. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 268. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述　针对 XML 存储器对象（XDA）数据的异步读取请求数。

用法　要计算每个异步请求读取的平均 XML 存储器对象数据页数，请使用以下公式:

```
pool_async_xda_reads / pool_async_xda_read_reqs
```

此平均数可帮助您确定每次与预取程序交互时完成的异步 I/O 量。

相关概念:

- 『XML 存储器对象概述』（《*管理指南: 计划*》）

相关参考:

- 第 220 页的『缓冲池活动监视元素』
- 第 239 页的『pool_async_data_read_reqs － 缓冲池异步读取请求数』
- 第 247 页的『pool_async_xda_reads － 缓冲池异步 XDA 数据读取数』

pool_async_xda_reads － 缓冲池异步 XDA 数据读取数

元素标识　pool_async_xda_reads

元素类型　计数器

表 269. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 270. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 指示从表空间容器（物理）中由异步引擎可派遣单元（EDU）对所有类型的表空间读取的 XML 存储器对象（XDA）数据页数。

用法 可将此元素与 pool_xda_p_reads 配合使用以计算对 XML 存储器对象数据页同步执行的物理读取数（即数据库管理器代理程序对 XML 数据执行的物理数据页读取数）。使用以下公式:

```
pool_xda_p_reads - pool_async_xda_reads
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 *num_ioservers* 配置参数（请参阅*管理指南*）时，此元素会很有帮助。

异步读取是由数据库管理器预取程序执行的。有关这些预取程序的信息，请参阅*管理指南*。

相关概念:

- 『XML 存储器对象概述』（《*管理指南: 计划*》）

相关参考:

- 第 220 页的『缓冲池活动监视元素』
- 第 234 页的『pool_async_data_reads – 缓冲池异步数据读取数』
- 第 250 页的『pool_xda_p_reads – 物理读取的缓冲池 XDA 数据页数』

pool_async_xda_writes – 缓冲池异步 XDA 数据写入数

元素标识	pool_async_xda_writes
元素类型	计数器

表 271. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 272. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

描述 异步页清除程序或预取程序将 XML 存储器对象（XDA）的缓冲池数据页物理写至磁盘的次数。预取程序可将脏页写至磁盘以便为要预取的页腾出空间。

用法 可将此元素与 pool_xda_writes 配合使用以计算对 XML 存储器对象数据页同步执行的物理写入请求数（即数据库管理器代理程序对 XML 数据执行的物理数据页写入数）。使用以下公式:

```
pool_xda_writes - pool_async_xda_writes
```

通过比较异步写入数与同步写入数的比率，可了解缓冲池页清除程序的执行情况。在调整 *num_iocleaners* 配置参数时，此比率会非常有用。

有关异步页清除程序的更多信息，请参阅*管理指南*。

相关概念:

- 『XML 存储器对象概述』（*《管理指南: 计划》*）

相关参考:

- 第 220 页的『缓冲池活动监视元素』
- 第 235 页的『pool_async_data_writes － 缓冲池异步数据写入数』
- 第 251 页的『pool_xda_writes － 缓冲池 XDA 数据写入数』

pool_xda_l_reads － 逻辑读取的缓冲池 XDA 数据页数

元素标识	pool_xda_l_reads
元素类型	计数器

表 273. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 274. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从缓冲池（逻辑）中对常规表空间和大型表空间请求的 XML 存储器对象（XDA）的数据页数。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 此计数包括数据处于下列情况时对数据的访问:

- 当数据库管理器需要处理页时索引页已经在缓冲池中
- 应读取到缓冲池中，数据库管理器才能处理页。

可将 *pool_xda_l_reads* 监视元素与 *pool_xda_p_reads*、*pool_data_l_reads* 和 *pool_data_p_reads* 一起使用并借助以下公式来计算缓冲池的数据页命中率:

```
 1 - ((pool_data_p_reads + pool_xda_p_reads) / (pool_data_l_reads
+ pool_xda_l_reads))
```

有关确定整体缓冲池命中率的信息，请参阅*缓冲池活动监视元素*。

增加缓冲池大小一般会改进命中率，但您会达到一个最优状态，而无法继续改进。从理论上说，如果能够分配大到足以存储整个数据库的缓冲池，则系统启动并运行后你可以得到 100% 的命中率。但在许多情况下这是不现实的。命中率的高低实际上取决于数据的大小以及访问数据的方式。如果数据库很大并且数据访问比较平均，则命中率将会很低。对于非常大的表来说，您几乎无能为力。在此情况下，应该将重点放在较小并且访问较为频繁的表以及索引上。如果可能的话，将它们指定给希望获取高命中率的各个缓冲池。

相关概念:

- 『XML 存储器对象概述』（*《管理指南: 计划》*）

相关参考:

- 第 220 页的『缓冲池活动监视元素』
- 第 222 页的『pool_data_l_reads – 缓冲池数据页逻辑读取数』
- 第 250 页的『pool_xda_p_reads – 物理读取的缓冲池 XDA 数据页数』

pool_xda_p_reads – 物理读取的缓冲池 XDA 数据页数

元素标识 pool_xda_p_reads

元素类型 计数器

表 275. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 276. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从表空间容器（物理）中对常规表空间和大型表空间读取的 XML 存储器对象（XDA）的数据页数。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 有关如何使用此元素的信息，请参阅 pool_xda_l_reads 和 pool_async_xda_reads。

相关概念:

- 『XML 存储器对象概述』（《*管理指南: 计划*》）

相关参考:

- 第 220 页的『缓冲池活动监视元素』
- 第 247 页的『pool_async_xda_reads – 缓冲池异步 XDA 数据读取数』
- 第 224 页的『pool_data_p_reads – 缓冲池数据页物理读取数』
- 第 249 页的『pool_xda_l_reads – 逻辑读取的缓冲池 XDA 数据页数』

pool_xda_writes – 缓冲池 XDA 数据写入数

元素标识 pool_xda_writes

元素类型 计数器

表 277. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 278. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

描述 指示 XML 存储器对象（XDA）的缓冲池数据页（物理）写至磁盘的次数。

用法 此监视元素帮助您评估通过增加数据库的可用缓冲池页数能否改进性能。对于包含 XML 数据的数据库，应考虑对于 XML 数据而言缓冲池页写入数与缓冲池页读取数的比率（使用 the pool_xda_writes 和 pool_xda_p_reads 监视元素）以及对于关系数据类型而言缓冲池页写入数与缓冲池页读取数的比率（使用 pool_data_writes 和 pool_data_p_reads 监视元素）。

有关如何使用此元素的信息，请参阅 pool_xda_l_reads 和 pool_xda_p_reads。

有关缓冲池大小的更多信息，请参阅*管理指南*。

相关概念:

- 『XML 存储器对象概述』（《*管理指南: 计划*》）

相关参考:

- 第 220 页的『缓冲池活动监视元素』
- 第 226 页的『pool_data_writes – 缓冲池数据页写入数』

- 第 249 页的『pool_xda_l_reads – 逻辑读取的缓冲池 XDA 数据页数』
- 第 250 页的『pool_xda_p_reads – 物理读取的缓冲池 XDA 数据页数』

pool_temp_xda_l_reads – 逻辑读取的缓冲池临时 XDA 数据页数

元素标识 pool_temp_xda_l_reads

元素类型 计数器

表 279. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 280. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从缓冲池（逻辑）中对临时表空间请求的 XML 存储器对象（XDA）的数据页数。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法

可将 *pool_temp_xda_l_reads* 监视元素与 *pool_temp_xda_p_reads*、*pool_temp_data_l_reads* 和 *pool_temp_data_p_reads* *一起使用，并借助以下公式来计算临时表空间中缓冲池的数据页命中率：* ：

```
 1 - ((pool_temp_data_p_reads + pool_temp_xda_p_reads) /
(pool_temp_data_l_reads + pool_temp_xda_l_reads))
```

有关确定整体缓冲池命中率的信息，请参阅*缓冲池活动监视元素*。

相关概念:

- 『XML 存储器对象概述』（*《管理指南: 计划》*）

相关参考:

- 第 220 页的『缓冲池活动监视元素』
- 第 224 页的『pool_temp_data_l_reads – 缓冲池临时数据逻辑读取数』
- 第 253 页的『pool_temp_xda_p_reads – 物理读取的缓冲池临时 XDA 数据页数』
- 第 249 页的『pool_xda_l_reads – 逻辑读取的缓冲池 XDA 数据页数』

pool_temp_xda_p_reads – 物理读取的缓冲池临时 XDA 数据页数

元素标识 pool_temp_xda_p_reads

元素类型 计数器

表 281. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 282. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-

描述 指示从表空间容器（物理）中对临时表空间读取的 XML 存储器对象（XDA）的数据页数。

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

用法 有关如何使用此元素的信息，请参阅 *pool_temp_xda_l_reads*。

相关概念:

- 『XML 存储器对象概述』（《*管理指南：计划*》）

相关参考:

- 第 220 页的『缓冲池活动监视元素』
- 第 225 页的『pool_temp_data_p_reads – 缓冲池临时数据物理读取数』
- 第 252 页的『pool_temp_xda_l_reads – 逻辑读取的缓冲池临时 XDA 数据页数』

xda_object_pages – XDA 对象页数

元素标识 xda_object_pages

元素类型 信息

表 283. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本

表 284. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

描述 XML 存储器对象（XDA）数据消耗的磁盘页数。

用法 此元素提供了一种机制，可用来查看特定表中的 XML 存储器对象（XDA）数据消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内 XML 存储器对象数据的增长率。

相关概念:

- 『XML 存储器对象概述』（《管理指南：计划》）

相关参考:

- 第 343 页的『data_object_pages － 数据对象页数』

动态缓冲池

bp_cur_buffsz － 缓冲池的当前大小:

元素标识 bp_cur_buffsz

元素类型 标尺

表 285. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

描述 当前缓冲池大小。

bp_new_buffsz － 新的缓冲池大小:

元素标识 bp_new_buffsz

元素类型 信息

表 286. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

描述 一旦重新启动数据库后缓冲池将更改至的大小。当以 DEFERRED 方式执行 ALTER BUFFERPOOL 语句时，在停止并重新启动数据库之前，缓冲池大小不会更改。

bp_pages_left_to_remove － 要除去的余下页数:

元素标识 bp_pages_left_to_remove

元素类型 标尺

表 287. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

描述 在完成缓冲池调整大小之前，缓冲池中要除去的余下页数。此项仅适用于以 IMMEDIATE 方式执行的 ALTER BUFFERPOOL 语句调用的缓冲池调整大小操作。

***bp_tbsp_use_count* － 映射至缓冲池的表空间数:**

元素标识 bp_tbsp_use_count

元素类型 标尺

表 288. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

描述 使用此缓冲池的表空间数。

非缓冲 I/O 活动

非缓冲 I/O 活动监视元素

下列元素提供有关未使用缓冲池的 I/O 活动的信息:
- direct_reads － 直接从数据库进行读取的读取操作数监视元素
- direct_writes － 直接写入数据库的写入操作数监视元素
- direct_read_reqs － 直接读取请求数监视元素
- direct_write_reqs － 直接写入请求数监视元素
- direct_read_time － 直接读取时间监视元素
- direct_write_time － 直接写入时间监视元素

direct_reads － 直接从数据库进行读取的读取操作数

元素标识 direct_reads

元素类型 计数器

表 289. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 290. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

描述 不使用缓冲池进行的读取操作数。

用法 使用以下公式来计算直接读取操作读取的平均扇区数:

```
direct_reads / direct_read_reqs
```

使用系统监视器跟踪 I/O 时，此元素可帮助您区分设备上的数据库 I/O 与非数据库 I/O。

直接读取操作是以单元为单位执行的，最小单元为 512 字节扇区。在下列情况下将使用它们:
- 读取 LONG VARCHAR 列
- 读取 LOB（大对象）列
- 执行备份

相关参考:

- 第 257 页的『direct_read_reqs － 直接读取请求数』
- 第 258 页的『direct_read_time － 直接读取时间』
- 第 256 页的『direct_writes － 直接写入数据库的写入操作数』

direct_writes － 直接写入数据库的写入操作数

元素标识 direct_writes

元素类型 计数器

表 291. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 292. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

描述 不使用缓冲池的写入操作数。

用法 使用以下公式来计算直接写入操作写入的平均扇区数:

```
direct_writes / direct_write_reqs
```

使用系统监视器跟踪 I/O 时，此元素可帮助您区分设备上的数据库 I/O 与非数据库 I/O。

直接写入操作是以单元为单位执行的，最小单元为 512 字节扇区。在下列情况下将使用它们:
- 写入 LONG VARCHAR 列
- 写入 LOB（大对象）列

- 执行恢复
- 执行装入。

相关参考:

- 第 257 页的『direct_write_reqs － 直接写入请求数』
- 第 259 页的『direct_write_time － 直接写入时间』
- 第 255 页的『direct_reads － 直接从数据库进行读取的读取操作数』

direct_read_reqs － 直接读取请求数

元素标识 direct_read_reqs

元素类型 计数器

表 293. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 294. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

描述 对一个或多个扇区的数据执行直接读取的请求数。

用法 使用以下公式来计算直接读取操作读取的平均扇区数:

```
direct_reads / direct_read_reqs
```

相关参考:

- 第 255 页的『direct_reads － 直接从数据库进行读取的读取操作数』
- 第 258 页的『direct_read_time － 直接读取时间』
- 第 257 页的『direct_write_reqs － 直接写入请求数』

direct_write_reqs － 直接写入请求数

元素标识 direct_write_reqs

元素类型 计数器

表 295. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池

表 295. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 296. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

描述 对一个或多个扇区的数据执行直接写入的请求数。

用法 使用以下公式来计算直接写入操作写入的平均扇区数:

```
direct_writes / direct_write_reqs
```

相关参考:

- 第 256 页的『direct_writes – 直接写入数据库的写入操作数』
- 第 259 页的『direct_write_time – 直接写入时间』
- 第 257 页的『direct_read_reqs – 直接读取请求数』

direct_read_time – 直接读取时间

元素标识 direct_read_time

元素类型 计数器

表 297. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 298. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

描述 执行直接读取所需的耗用时间(以毫秒计)。

用法 使用以下公式来计算每扇区平均直接读取时间:

```
direct_read_time / direct_reads
```

如果平均时间很长，则指示可能存在 I/O 冲突。

相关参考:

- 第 255 页的『direct_reads – 直接从数据库进行读取的读取操作数』
- 第 257 页的『direct_read_reqs – 直接读取请求数』
- 第 259 页的『direct_write_time – 直接写入时间』

direct_write_time – 直接写入时间

元素标识 direct_write_time

元素类型 计数器

表 299. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 300. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

描述 执行直接写入所需的耗用时间（以毫秒计）。

用法 使用以下公式来计算每扇区平均直接写入时间:

```
direct_write_time / direct_writes
```

如果平均时间很长，则指示可能存在 I/O 冲突。

相关参考:

- 第 256 页的『direct_writes – 直接写入数据库的写入操作数』
- 第 257 页的『direct_write_reqs – 直接写入请求数』
- 第 258 页的『direct_read_time – 直接读取时间』

目录高速缓存

目录高速缓存监视元素

目录高速缓存存储:

- 表、视图和别名的表描述符。描述符以压缩内部格式存储有关表、视图或别名的信息。当 SQL 语句引用表时，它会导致将表描述符插入到高速缓存中，这样引用同一个表的后续 SQL 语句可使用该描述符从而避免从磁盘读取。（操作在编译 SQL 语句时引用表描述符。）

- 数据库权限信息。在处理 BIND、CONNECT、CREATE 和 LOAD 之类的语句时将访问数据库权限信息。当语句引用数据库权限信息时，可从目录高速缓存（而不是磁盘）访问对同一个用户或组引用数据库权限信息的后续操作。
- 针对例程（如用户定义的函数和存储过程）的执行特权。当事务对特定例程引用执行特权时，引用同一个例程的后续操作可从目录高速缓存（而不是磁盘）检索该信息。

下列数据库系统监视器元素可用于目录高速缓存:
- cat_cache_lookups – 目录高速缓存查询数监视元素
- cat_cache_inserts – 目录高速缓存插入数监视元素
- cat_cache_overflows – 目录高速缓存溢出数监视元素
- cat_cache_size_top – 目录高速缓存高水位标记监视元素

cat_cache_lookups – 目录高速缓存查询数

元素标识 cat_cache_lookups

元素类型 计数器

表 301. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 302. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 为获取表描述符信息或权限信息而引用目录高速缓存的次数。

用法 此元素包括对目录高速缓存的成功访问和不成功访问。每当出现下列情况，就会引用目录高速缓存:

- 在编译 SQL 语句期间处理表、视图或别名
- 访问数据库权限信息
- 在编译 SQL 语句期间处理例程

要计算目录高速缓存命中率，请使用以下公式:

```
(1 - (cat_cache_inserts / cat_cache_lookups))
```

来指示目录高速缓存避免目录访问的效果如何。如果比率很高（超过 0.8），则表示高速缓存确实起到作用。如果比率偏低，则表示应提高 *catalogcache_sz*。首次连接至数据库之后应该有较大的比率。

执行涉及表、视图或别名的数据定义语言（DDL）SQL 语句会从目录高速缓存中除去该对象的表描述符信息，从而导致这些信息在下一次引用时重新插入。

此外，用于数据库权限和例程执行特权的 GRANT 和 REVOKE 语句会从目录高速缓存中除去主题权限信息。因此，大量使用 DDL 语句和 GRANT/REVOKE 语句也会提高该比率。

有关目录高速缓存大小配置参数的更多信息，请参阅*管理指南*。

相关参考:

- 第 261 页的『cat_cache_inserts – 目录高速缓存插入数』
- 第 261 页的『cat_cache_overflows – 目录高速缓存溢出数』
- 第 262 页的『cat_cache_size_top – 目录高速缓存高水位标记』
- 第 359 页的『ddl_sql_stmts – 数据定义语言（DDL）SQL 语句数』

cat_cache_inserts – 目录高速缓存插入数

元素标识 cat_cache_inserts

元素类型 计数器

表 303. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 304. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 系统尝试将表描述符或权限信息插入到目录高速缓存中的次数。

用法 通过与“目录高速缓存查询”一起使用，可借助以下公式来计算目录高速缓存命中率:

```
1 -（目录高速缓存插入数 / 目录高速缓存查询数）
```

有关使用此元素的更多信息，请参阅 cat_cache_lookups。

相关参考:

- 第 260 页的『cat_cache_lookups – 目录高速缓存查询数』
- 第 261 页的『cat_cache_overflows – 目录高速缓存溢出数』
- 第 262 页的『cat_cache_size_top – 目录高速缓存高水位标记』

cat_cache_overflows – 目录高速缓存溢出数

元素标识 cat_cache_overflows

元素类型 计数器

表 305. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 306. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 目录高速缓存溢出其分配内存边界的次数。

用法 将此元素与 *cat_cache_size_top* 配合使用来确定是否需要增加目录高速缓存的大小以避免溢出。

目录高速缓存空间是通过除去表、视图或别名的表描述符信息或当前未被任何事务使用的权限信息来回收的。

如果 *cat_cache_overflows* 很大，则相对工作负载而言目录高速缓存可能会太小。扩大目录高速缓存可以改进性能。如果工作负载包括的事务将编译大量 SQL 语句，而这些语句又引用单个工作单元中的多个表、视图、别名、用户定义的函数或存储过程，则在单个事务中编译较少的 SQL 语句可以改进目录高速缓存的性能。或者，如果工作负载包括绑定包含许多 SQL 语句的程序包，而这些语句又引用多个表、视图、别名、用户定义的函数或存储过程，则可以尝试分割程序包以使这些程序包包括较少的 SQL 语句从而改进性能。

相关参考:

- 第 260 页的『cat_cache_lookups – 目录高速缓存查询数』
- 第 261 页的『cat_cache_inserts – 目录高速缓存插入数』
- 第 262 页的『cat_cache_size_top – 目录高速缓存高水位标记』

cat_cache_size_top – 目录高速缓存高水位标记

元素标识 cat_cache_size_top

元素类型 水位标记

表 307. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 308. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 目录高速缓存达到的最大大小。

用法 此元素指示对激活后的数据库运行工作负载所需的目录高速缓存最多字节数。

如果目录高速缓存溢出，则表示此元素包含溢出期间目录高速缓存达到的最大大小。检查目录高速缓存溢出数以确定是否存在此情况。

可通过以下公式来确定工作负载需要的目录高速缓存最小大小:

```
最大目录高速缓存大小 / 4096
```

通过将结果四舍五入为整数，指示为避免溢出目录高速缓存最少需要的页数（每页 4K 字节）。

相关参考:

- 第 261 页的『cat_cache_overflows – 目录高速缓存溢出数』

程序包高速缓存

程序包高速缓存监视元素

必要时，执行动态和静态 SQL 语句所需的程序包和段信息将放在程序包高速缓存中。每当要执行动态或静态语句时，就需要此信息。程序包高速缓存存在于数据库级别。这表示处于相同环境的代理程序可以共享另一代理程序的成果。对于静态 SQL 语句，这可能意味着避免目录访问。对于动态 SQL 语句，这可能意味着编译成本。

下列数据库系统监视元素用于程序包高速缓存:

- pkg_cache_lookups – 程序包高速缓存查询数监视元素
- pkg_cache_inserts – 程序包高速缓存插入数监视元素
- pkg_cache_num_overflows – 程序包高速缓存溢出数监视元素
- pkg_cache_size_top – 程序包高速缓存高水位标记监视元素
- appl_section_lookups – 节查询数监视元素
- appl_section_inserts – 节插入数监视元素

pkg_cache_lookups – 程序包高速缓存查询数

元素标识	pkg_cache_lookups
元素类型	计数器

表 309. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 310. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 应用程序在程序包高速缓存中查找某个段或程序包的次数。在数据库级别，这指示自数据库启动或监视器数据复位后的整体引用数。

注: 此计数器包括该段已装入到高速缓存中的情况和该段必须装入到高速缓存中的情况。

在代理程序要与不同应用程序相关联的集中器环境中，如果新的代理程序在本地存储器中没有必需的段或程序包可用，则可能需要附加程序包高速缓存查询。

用法 要计算程序包高速缓存命中率，请使用以下公式:

```
1 - (程序包高速缓存插入数 / 程序包高速缓存查询数)
```

程序包高速缓存命中率显示程序包高速缓存的使用是否有效率。如果命中率很高（超过 0.8），则表示高速缓存确实起到作用。如果命中率偏低，则可能指示应该增加程序包高速缓存。

您需要使用程序包高速缓存大小进行试验，以找出 *pckcachesz* 配置参数的最优数字。例如，如果降低高速缓存的大小时 *pkg_cache_inserts* 元素中的值没有增加，则可以使用更小的程序包高速缓存大小。降低程序包高速缓存大小将释放系统资源以供其他任务使用。如果增加程序包高速缓存大小使得 pkg_cache_inserts 数降低，则可以通过这样做来改进整体系统性能。在满工作负载条件下能够最好地完成此实验。

可将此元素与 *ddl_sql_stmts* 配合使用以确定执行 DDL 语句是否会影响程序包高速缓存的性能。执行 DDL 语句时，动态 SQL 语句的各段可能会变得无效。下次使用时系统会对无效段进行隐式编译。执行 DDL 语句可能会使许多段变得无效，并且编译这些段时产生的其他开销会严重影响性能。在此情况下，程序包高速缓存命中率会影响无效段的隐式重新编译。但它不会影响将新段插入到高速缓存中，所以增加程序包高速缓存大小不会改进整体性能。您可能会发现在满负载环境中工作之前独自为应用程序调整高速缓存会更加清晰明了。

在决定采取什么操作之前，需要确定 DDL 语句在程序包高速缓存命中率的值中所起的作用。如果 DDL 语句很少出现，则可通过增加高速缓存大小来改进性能。如果 DDL 语句频繁出现，则可能需要通过限制 DDL 语句的使用（将 DDL 语句的可能使用时间限制为特定时间段）来改进性能。

static_sql_stmts 和 *dynamic_sql_stmts* 计数可用来帮助提供有关要高速缓存的段的数量和类型的信息。

有关程序包高速缓存大小（*pckcachesz*）配置参数的更多信息，请参阅*管理指南*。

注: 您可能想要在数据库级别使用此信息来计算每个应用程序的平均程序包高速缓存命中率。您应该在应用程序级别查看此信息以找出给定应用程序的精确程序包高速缓存命中率。为了满足很少执行的应用程序的高速缓存需求而增加程序包高速缓存大小是不值得的。

相关参考:

- 第 265 页的『pkg_cache_inserts － 程序包高速缓存插入数』
- 第 354 页的『static_sql_stmts － 尝试的静态 SQL 语句数』
- 第 354 页的『dynamic_sql_stmts － 尝试的动态 SQL 语句数』
- 第 359 页的『ddl_sql_stmts － 数据定义语言（DDL）SQL 语句数』

pkg_cache_inserts － 程序包高速缓存插入数

元素标识 pkg_cache_inserts

元素类型 计数器

表 311. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 312. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 请求段不可用并且必须装入到程序包高速缓存中的总次数。此计数包括系统执行的所有隐式编译。

用法 通过与“程序包高速缓存查询数”一起使用，可借助以下公式来计算程序包高速缓存命中率:

```
1 - (程序包高速缓存插入数 / 程序包高速缓存查询数)
```

有关使用此元素的更多信息，请参阅 pkg_cache_lookups。

相关参考:

- 第 263 页的『pkg_cache_lookups － 程序包高速缓存查询数』

pkg_cache_num_overflows － 程序包高速缓存溢出数

元素标识 pkg_cache_num_overflows

元素类型 计数器

表 313. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 314. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 程序包高速缓存溢出其分配内存边界的次数。

用法 将此元素与 pkg_cache_size_top 配合使用来确定是否需要增加程序包高速缓存的大小以避免溢出。

相关参考:

• 第 265 页的『pkg_cache_inserts – 程序包高速缓存插入数』
• 第 359 页的『ddl_sql_stmts – 数据定义语言（DDL）SQL 语句数』
• 第 354 页的『dynamic_sql_stmts – 尝试的动态 SQL 语句数』
• 第 354 页的『static_sql_stmts – 尝试的静态 SQL 语句数』

pkg_cache_size_top – 程序包高速缓存高水位标记

元素标识 pkg_cache_size_top

元素类型 水位标记

表 315. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 316. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 程序包高速缓存达到的最大大小。

用法 此元素指示对激活后的数据库运行工作负载所需的程序包高速缓存最多字节数。

如果程序包高速缓存溢出，则表示此元素包含溢出期间程序包高速缓存达到的最大大小。查看程序包高速缓存溢出数以确定是否出现这种情况。

可通过以下公式来确定工作负载需要的程序包高速缓存最小大小:

```
最大程序包高速缓存大小 / 4096
```

通过将结果四舍五入为整数，指示为避免溢出程序包高速缓存最少需要的页数（每页 4K 字节）。

相关参考:

• 第 265 页的『pkg_cache_num_overflows – 程序包高速缓存溢出数』

appl_section_lookups – 段查询

元素标识 appl_section_lookups

元素类型 计数器

表 317. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 318. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

表 318. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

描述 应用程序从其 SQL 工作区进行的对 SQL 段的查询。

用法 每个代理程序可以访问保留任何可执行段的工作副本的唯一 SQL 工作区。在分区数据库中，此工作区由所有非 SMP 代理程序共享。在其他环境中并且具有 SMP 代理程序的情况下，每个代理程序有自己的唯一 SQL 工作区。

此计数器指示应用程序的代理程序访问 SQL 工作区的次数。它是为此应用程序工作的代理程序对所有 SQL 工作堆的所有查询的累积总计。

可将此元素与 *appl_section_inserts* 一起使用来调整用于 SQL 工作区的堆的大小。在分区数据库中，此大小由 *app_ctl_heap_sz* 配置参数控制。其他数据库环境中的 SQL 工作区大小使用 *applheapsz* 配置参数。在所有环境中，SMP 代理程序的 SQL 工作区大小都由 *applheapsz* 控制。

相关参考:

- 『app_ctl_heap_sz – 应用程序控制堆大小配置参数』(《*性能指南*》)
- 第 267 页的『appl_section_inserts – 段插入数监视元素』
- 『applheapsz – 应用程序堆大小配置参数』(《*性能指南*》)
- 第 265 页的『pkg_cache_inserts – 程序包高速缓存插入数』
- 第 263 页的『pkg_cache_lookups – 程序包高速缓存查询数』

appl_section_inserts – 段插入数监视元素

元素标识 appl_section_inserts

元素类型 计数器

表 319. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

表 320. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 应用程序从其 SQL 工作区插入 SQL 段的次数。

用法 任何可执行段的工作副本存储在唯一 SQL 工作区中。这是出现副本不可用并且必须插入的情况的计数。

有关使用段的更多信息，请参阅 *appl_section_lookups*。

相关参考:

- 第 266 页的『appl_section_lookups – 段查询』
- 第 265 页的『pkg_cache_inserts – 程序包高速缓存插入数』

- 第 263 页的『pkg_cache_lookups – 程序包高速缓存查询数』

SQL 工作空间

SQL 工作空间监视元素

如果应用程序在执行动态或静态 SQL 语句时需要某些段，则这些段将根据需要放在共享工作空间或专用工作空间中。共享工作空间存在于应用程序级别，由一个或多个应用程序共享。专用工作空间存在于代理程序级别，并且每个代理程序只有一个相关联的专用工作空间。

因为共享工作空间是在多个应用程序间共享的，所以处于相同环境的应用程序可以共享另一代理程序的工作成果。已经实现的好处包括设置和初始化成本。

下列数据库系统监视元素用于 SQL 工作空间:
- shr_workspace_size_top – 最大共享工作空间大小监视元素
- shr_workspace_num_overflows – 共享工作空间溢出数监视元素
- shr_workspace_section_lookups – 共享工作空间节查询数监视元素
- shr_workspace_section_inserts – 共享工作空间节插入数监视元素
- priv_workspace_size_top – 最大专用工作空间大小监视元素
- priv_workspace_num_overflows – 专用工作空间溢出数监视元素
- priv_workspace_section_lookups – 专用工作空间节查询数监视元素
- priv_workspace_section_inserts – 专用工作空间节插入数监视元素

shr_workspace_size_top – 最大共享工作空间大小

元素标识 shr_workspace_size_top

元素类型 水位标记

表 321. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

表 322. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 共享工作空间达到的最大大小。

用法 此元素指示对激活后的数据库运行工作负载所需的共享工作空间最多字节数。在数据库级别，此项是所有共享工作空间达到的最大大小。 在应用程序级别，此项是当前应用程序使用的共享工作空间的最大大小。

如果共享工作空间溢出，则表示此元素包含溢出期间共享工作空间达到的最大大小。检查共享工作空间溢出数以确定是否存在此情况。

工作空间溢出时，将会临时从应用程序共享内存的其他实体借出内存。这可能导致这些实体出现内存不足错误，也可能导致性能下降。可通过增加 APP_CTL_HEAP_SZ 来降低溢出的机率。

相关参考:

- 第 269 页的『shr_workspace_num_overflows – 共享工作空间溢出数』

shr_workspace_num_overflows – 共享工作空间溢出数

元素标识 shr_workspace_num_overflows

元素类型 计数器

表 323. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 324. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 共享工作空间溢出其分配内存边界的次数。

用法 将此元素与 shr_workspace_size_top 配合使用来确定是否需要增加共享工作空间的大小以避免溢出。共享工作空间的溢出可能导致性能下降，以及从应用程序共享内存分配的其他堆出现内存不足错误。

在数据库级别，报告的元素将来自报告具有最大共享工作空间大小的元素的共享工作空间。在应用程序级别，此项是当前应用程序使用的工作空间的溢出数。

相关参考:

- 第 268 页的『shr_workspace_size_top – 最大共享工作空间大小』

shr_workspace_section_lookups – 共享工作空间段查询数

元素标识 shr_workspace_section_lookups

元素类型 计数器

表 325. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 326. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 应用程序在共享工作空间中对 SQL 段进行查询的次数。

用法 每个应用程序都可以访问共享工作空间，该工作空间中保留了可执行段的工作副本。

此计数器指示为找到应用程序的特定段而访问共享工作空间的次数。在数据库级别，此项是针对数据库的所有共享工作空间中的每个应用程序进行的所有查询的累积总数。在应用程序级别，此项是针对此应用程序的共享工作空间中的所有段的所有查询的累积总数。

可将此元素与“共享工作空间段插入数”一起使用来调整共享工作空间的大小。共享工作空间的大小由 app_ctl_heap_sz 配置参数控制。

相关参考:

• 第 270 页的『shr_workspace_section_inserts － 共享工作空间段插入数』

shr_workspace_section_inserts － 共享工作空间段插入数

元素标识 shr_workspace_section_inserts

元素类型 计数器

表 327. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 328. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 应用程序将 SQL 段插入到共享工作空间中的次数。

用法 可执行段的工作副本存储在共享工作空间中。此计数器指示副本何时不可用并且必须插入。

在数据库级别，此项是针对数据库的所有共享工作空间中的每个应用程序进行的所有插入的累积总数。在应用程序级别，此项是针对此应用程序的共享工作空间中的所有段的所有插入的累积总数。

相关参考:

• 第 269 页的『shr_workspace_section_lookups － 共享工作空间段查询数』

priv_workspace_size_top – 最大专用工作空间大小

元素标识 priv_workspace_size_top

元素类型 水位标记

表 329. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

表 330. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 专用工作空间达到的最大大小。

用法 每个代理程序都有专用工作空间，它所服务的应用程序对该工作空间具有访问权。此元素指示专用工作空间中提供服务的任何代理程序所需的最多字节数。在数据库级别，此项是连接至当前数据库的所有代理程序在所有专用工作空间中必需的最多字节数。 在应用程序级别，此项是为当前应用程序提供服务的所有代理程序的专用空间中的最大大小。

专用工作空间溢出时，将会临时从代理程序专用内存的其他实体借出内存。这可能导致这些实体出现内存不足错误，也可能导致性能下降。可通过增加 APPLHEAPSZ 来降低溢出的机率。

相关参考:

- 第 271 页的『priv_workspace_num_overflows – 专用工作空间溢出数』

priv_workspace_num_overflows – 专用工作空间溢出数

元素标识 priv_workspace_num_overflows

元素类型 计数器

表 331. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 332. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 专用工作空间溢出其分配内存边界的次数。

用法 将此元素与 priv_workspace_size_top 配合使用来确定专用工作空间是否需要增

加大小以避免溢出。专用工作空间的溢出可能导致性能下降，以及从代理程序专用内存分配的其他堆出现内存不足错误。

在数据库级别，报告的元素将来自报告为具有相同最大专用工作空间大小的元素的专用工作空间。在应用程序级别，此项是为当前应用程序提供服务的每个代理程序的工作空间的溢出数。

相关参考:

- 第 271 页的『priv_workspace_size_top － 最大专用工作空间大小』

priv_workspace_section_lookups － 专用工作空间段查询数

元素标识 priv_workspace_section_lookups

元素类型 计数器

表 333. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 334. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 应用程序在其代理程序的专用工作空间中对 SQL 段进行查询的次数。

用法 每个应用程序都可以访问为其工作的代理程序的专用工作空间。

此计数器指示为找到应用程序的特定段而访问专用工作空间的次数。在数据库级别，此项是针对数据库的所有专用工作空间中的每个应用程序进行的所有查询的累积总数。在应用程序级别，此项是针对此应用程序的专用工作空间中的所有段的所有查询的累积总数。

可将此元素与“专用工作空间段插入数”一起使用来调整专用工作空间的大小。专用工作空间的大小由 applheapsz 配置参数控制。

相关参考:

- 第 272 页的『priv_workspace_section_inserts － 专用工作空间段插入数』

priv_workspace_section_inserts － 专用工作空间段插入数

元素标识 priv_workspace_section_inserts

元素类型 计数器

表 335. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 336. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 应用程序在专用工作空间中插入 SQL 段的次数。

用法 可执行段的工作副本存储在专用工作空间中。

此计数器指示副本何时不可用并且必须插入。在数据库级别，此项是针对数据库的所有专用工作空间中的每个应用程序进行的所有插入的累积总数。在应用程序级别，此项是针对此应用程序的专用工作空间中的所有段的所有插入的累积总数。

在代理程序要与不同应用程序相关联的集中器环境中，如果新的代理程序在专用工作空间中没有必需的段可用，则可能需要附加专用工作空间插入。

相关参考:

- 第 272 页的『priv_workspace_section_lookups － 专用工作空间段查询数』

数据库堆

数据库堆监视元素

下列数据库系统监视元素用于数据库堆:

- db_heap_top － 分配的最大数据库堆监视元素

db_heap_top － 分配的最大数据库堆

元素标识 db_heap_top

元素类型 水位标记

表 337. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 338. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 此元素将保留以获取 DB2 版本兼容性。它现在量度内存使用情况，但并非由数据库堆专用。

日志记录

记录监视元素

下列数据库系统监视器元素用于记录:

- sec_log_used_top – 已用的最大辅助日志空间监视元素
- tot_log_used_top – 已用的最大总日志空间监视元素
- sec_logs_allocated – 当前分配的辅助日志数监视元素
- log_reads – 读取的日志页数监视元素
- log_writes – 写入的日志页数监视元素
- uow_log_space_used – 使用的工作单元日志空间监视元素
- total_log_used – 使用的总日志空间监视元素
- total_log_available – 可用的总日志数监视元素
- log_held_by_dirty_pages – 脏页所占的日志空间量监视元素log_held_by_dirty_pages – 脏页所占的日志空间量监视元素
- log_to_redo_for_recovery – 要为恢复而重做的日志数监视元素log_to_redo_for_recovery – 要为恢复而重做的日志数监视元素
- log_write_time – 日志写入时间监视元素log_write_time – 日志写入时间监视元素
- log_read_time – 日志读取时间监视元素log_read_time – 日志读取时间监视元素
- num_log_write_io – 日志写入数监视元素num_log_write_io – 日志写入数监视元素
- num_log_read_io – 日志读取数监视元素num_log_read_io – 日志读取数监视元素
- num_log_part_page_io – 部分日志页写入数监视元素num_log_part_page_io – 部分日志页写入数监视元素
- num_log_buffer_full – 已满的日志缓冲区数监视元素num_log_buffer_full – 已满的日志缓冲区数监视元素
- num_log_data_found_in_buffer – 在缓冲区中找到的日志数据数监视元素num_log_data_found_in_buffer – 在缓冲区中找到的日志数据数监视元素
- first_active_log – 首个活动日志文件号监视元素first_active_log – 首个活动日志文件号监视元素
- last_active_log – 最后一个活动日志文件号监视元素last_active_log – 最后一个活动日志文件号监视元素
- current_active_log – 当前活动日志文件号监视元素current_active_log – 当前活动日志文件号监视元素
- current_archive_log – 当前归档日志文件号监视元素current_archive_log – 当前归档日志文件号监视元素

sec_log_used_top – 使用的最大辅助日志空间

元素标识 sec_log_used_top

元素类型 水位标记

表 339. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 340. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 使用的最大辅助日志空间（以字节计）。

用法 可将此元素与 *sec_logs_allocated* 和 *tot_log_used_top* 配合使用以显示当前对辅助日志的依赖性。如果此值很高，则可能需要更大的日志文件或者更多的主日志文件，又或是在应用程序中更频繁地使用 COMMIT 语句。

因此，可能需要调整下列配置参数:
- logfilsiz
- logprimary
- logsecond
- logretain

如果数据库没有任何辅助日志文件，则该值将为零。如果未定义辅助日志文件，则将出现此情况。

有关更多信息，请参阅*管理指南*。

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

相关参考:

- 第 275 页的『tot_log_used_top － 使用的最大总日志空间』
- 第 277 页的『uow_log_space_used － 使用的工作单元日志空间』
- 第 276 页的『sec_logs_allocated － 目前分配的辅助日志数』

tot_log_used_top － 使用的最大总日志空间

元素标识 tot_log_used_top

元素类型 水位标记

表 341. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 342. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 使用的最大总日志空间（以字节计）。

用法 可使用此元素来帮助评估分配的主日志空间量。将此元素的值与分配的主日志空间量进行比较可帮助您评估配置参数设置。可借助以下公式计算主日志空间分配:

```
logprimary x logfilsiz x 4096（请参阅以下注释）
```

可将此元素与 *sec_log_used_top* 和 *sec_logs_allocated* 一起使用来显示当前对辅助日志的依赖性。

此值同时包括在主日志文件和辅助日志文件中使用的空间。

可能需要调整下列配置参数:

- logfilsiz
- logprimary
- logsecond

有关更多信息，请参阅*管理指南*。

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

相关参考:

- 第 277 页的『uow_log_space_used － 使用的工作单元日志空间』
- 第 276 页的『sec_logs_allocated － 目前分配的辅助日志数』
- 第 274 页的『sec_log_used_top － 使用的最大辅助日志空间』

sec_logs_allocated － 目前分配的辅助日志数

元素标识 sec_logs_allocated

元素类型 标尺

表 343. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 当前用于数据库的辅助日志文件总数。

用法 可将此元素与 *sec_log_used_top* 和 *tot_log_used_top* 一起使用来显示当前对辅助日志的依赖性。如果此值一直很高，则可能需要更大的日志文件或者更多的主日志文件，又或是在应用程序中更频繁地使用 COMMIT 语句。

因此，可能需要调整下列配置参数:

- logfilsiz
- logprimary
- logsecond
- logretain

有关更多信息，请参阅*管理指南*。

相关参考:

- 第 277 页的『uow_log_space_used － 使用的工作单元日志空间』
- 第 274 页的『sec_log_used_top － 使用的最大辅助日志空间』
- 第 275 页的『tot_log_used_top － 使用的最大总日志空间』

log_reads － 读取的日志页数

元素标识 log_reads

元素类型 计数器

表 344. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 345. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 记录器从磁盘读取的日志页数。

用法 可将此元素与操作系统监视器配合使用来确定设备上可归因于数据库活动的 I/O 量。

相关参考:

- 第 277 页的『log_writes － 写入的日志页数』

log_writes － 写入的日志页数

元素标识 log_writes

元素类型 计数器

表 346. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 347. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 记录器写至磁盘的日志页数。

用法 可将此元素与操作系统监视器配合使用来确定设备上可归因于数据库活动的 I/O 量。

注: 当日志页写至磁盘时，最后一页可能未满。在这种情况下，部分日志页保留在日志缓冲区中，而其他日志记录将写至页。因此记录器可能会多次将日志页写至磁盘。不应使用此元素来量度 DB2 生成的页数。

相关参考:

- 第 276 页的『log_reads － 读取的日志页数』

uow_log_space_used － 使用的工作单元日志空间

元素标识 uow_log_space_used

元素类型 标尺

表 348. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元

表 349. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	-

描述 被监视应用程序的当前工作单元中使用的日志空间量（以字节计）。

用法 可使用此元素来了解工作单元级别的记录需求。

相关参考:

- 第 276 页的『sec_logs_allocated － 目前分配的辅助日志数』
- 第 274 页的『sec_log_used_top － 使用的最大辅助日志空间』
- 第 275 页的『tot_log_used_top － 使用的最大总日志空间』

total_log_used － 使用的总日志空间

元素标识 total_log_used

元素类型 标尺

表 350. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 当前在数据库中使用的总活动日志空间量（以字节计）。

用法 将此元素与 total_log_available 一起使用来确定是否需要调整下列配置参数以避免用完日志空间:

- logfilsiz
- logprimary
- logsecond

有关更多信息，请参阅*管理指南*。

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

相关参考:

- 第 277 页的『uow_log_space_used － 使用的工作单元日志空间』
- 第 276 页的『sec_logs_allocated － 目前分配的辅助日志数』
- 第 275 页的『tot_log_used_top － 使用的最大总日志空间』
- 第 162 页的『appl_id_oldest_xact － 带有最旧事务的应用程序』

total_log_available － 可用的总日志量

元素标识 total_log_available

元素类型 标尺

表 351. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 数据库中未被未落实事务使用的活动日志空间量（以字节计）。

用法 将此元素与 total_log_used 一起使用来确定是否需要调整下列配置参数以避免用完日志空间:

- logfilsiz
- logprimary
- logsecond

如果 total_log_available 降低至 0，则将返回 SQL0964N。您可能需要增加以上配置参数，或通过 COMMIT、ROLLBACK 或 FORCE APPLICATION 结束最旧的事务。

如果 logsecond 设置为 -1，则此元素将包含 SQLM_LOGSPACE_INFINITE。

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

相关参考:

- 第 277 页的『uow_log_space_used – 使用的工作单元日志空间』
- 第 276 页的『sec_logs_allocated – 目前分配的辅助日志数』
- 第 278 页的『total_log_used – 使用的总日志空间』
- 第 162 页的『appl_id_oldest_xact – 带有最旧事务的应用程序』

log_held_by_dirty_pages – 脏页占用的日志空间量

元素标识 log_held_by_dirty_pages

元素类型 水位标记

表 352. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 353. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 对应数据库中的最旧脏页与活动日志顶部之间的差的日志量（以字节计）。

用法 获取快照时，将根据该快照的时间上的条件计算此值。

使用此元素来评估对缓冲池中的旧页进行页清除的效果。

缓冲池中的旧页清除由 *softmax* 数据库配置参数控制。如果页清除有效果，则 *log_held_by_dirty_pages* 应小于或大致等于:

```
(softmax / 100)  *  logfilsiz * 4096
```

如果不是这样，则增加页清除程序的数目（*num_iocleaners*）配置参数。

如果符合条件并且期望脏页占用的日志少一些，则降低 *softmax* 配置参数。

相关参考:

- 第 280 页的『log_to_redo_for_recovery – 要为恢复重做的日志量』
- 『logfilsiz – 日志文件大小配置参数』（*《性能指南》*）
- 『num_iocleaners – 异步页清除程序的数目配置参数』（*《性能指南》*）
- 第 240 页的『pool_drty_pg_steal_clns – 触发的缓冲池牺牲页清除程序数』
- 第 242 页的『pool_drty_pg_thrsh_clns – 触发的缓冲池阈值清除程序数』
- 第 240 页的『pool_lsn_gap_clns – 触发的缓冲池日志空间清除程序数』
- 『softmax – 恢复范围和软检查点时间间隔配置参数』（*《性能指南》*）

log_to_redo_for_recovery – 要为恢复重做的日志量

元素标识 log_to_redo_for_recovery

元素类型 水位标记

表 354. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 355. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 必须为崩溃恢复重做的日志量（以字节计）。

用法 获取快照时，将根据该快照的时间上的条件计算此值。如果值较大，则指示系统崩溃后的恢复时间较长。如果值看起来过大，则检查 *log_held_by_dirty_pages* 监视元素以了解是否需要调整页清除。还应检查是否存在任何长时间运行并且需要终止的事务。

相关参考:

- 第 279 页的『log_held_by_dirty_pages – 脏页占用的日志空间量』

log_write_time – 日志写入时间

元素标识 log_write_time

元素类型 时间

表 356. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 357. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 记录器将日志数据写至磁盘的总耗用时间。

用法 将此元素与 *log_writes* 和 *num_log_write_io* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

相关参考:

- 第 277 页的『log_writes － 写入的日志页数』
- 第 281 页的『num_log_write_io － 日志写入次数』

log_read_time － 日志读取时间

元素标识 log_read_time

元素类型 时间

表 358. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 359. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 记录器从磁盘读取日志数据的总耗用时间。

用法 将此元素与 *log_reads*、*num_log_read_io* 和 *num_log_data_found_in_buffer* 元素一起使用来确定以下情况是否属实:

- 当前磁盘对于记录操作而言是够用的。
- 日志缓冲区大小够用。

相关参考:

- 第 276 页的『log_reads － 读取的日志页数』
- 第 283 页的『num_log_data_found_in_buffer － 在缓冲区中找到日志数据的次数』
- 第 282 页的『num_log_read_io － 日志读取数』

num_log_write_io － 日志写入次数

元素标识 num_log_write_io

元素类型 计数器

表 360. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 361. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 记录器为了将日志数据写至磁盘而发出的 I/O 请求数。

用法 将此元素与 *log_writes* 和 *log_write_time* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

相关参考:

- 第 280 页的『log_write_time － 日志写入时间』
- 第 277 页的『log_writes － 写入的日志页数』

num_log_read_io － 日志读取数

元素标识	num_log_read_io
元素类型	计数器

表 362. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 363. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 记录器为了从磁盘读取日志数据而发出的 I/O 请求数。

用法 将此元素与 *log_reads* 和 *log_read_time* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

相关参考:

- 第 281 页的『log_read_time － 日志读取时间』
- 第 276 页的『log_reads － 读取的日志页数』

num_log_part_page_io － 部分日志页写入数

元素标识	num_log_part_page_io
元素类型	计数器

表 364. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 365. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 记录器为了将部分日志数据写至磁盘而发出的 I/O 请求数。

用法 将此元素与 *log_writes*、*log_write_time* 和 *num_log_write_io* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

相关参考:

- 第 280 页的『log_write_time – 日志写入时间』
- 第 277 页的『log_writes – 写入的日志页数』
- 第 281 页的『num_log_write_io – 日志写入次数』

num_log_buffer_full – 变满的日志缓冲区的数目

元素标识 num_log_buffer_full

元素类型 计数器

表 366. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

描述 在将日志记录复制至日志缓冲区时，代理程序必须等待日志数据写入磁盘的次数。每个代理程序每一次发生这种情况时，此值都会递增。例如，如果两个代理程序尝试在缓冲区变满时复制日志数据，则此值将加上 2。

用法 使用此元素来确定是否需要增加 LOGBUFSZ 数据库配置参数的大小。

相关参考:

- 『logbufsz – 日志缓冲区大小配置参数』（*《性能指南》*）

num_log_data_found_in_buffer – 在缓冲区中找到日志数据的次数

元素标识 num_log_data_found_in_buffer

元素类型 计数器

表 367. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 368. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 代理程序从缓冲区读取日志数据的次数。

从缓冲区读取日志数据比从磁盘读取数据要好一些，这是因为从磁盘读取时速度较慢。

用法 将此元素与 *num_log_read_io* 元素一起使用来确定是否需要增加 LOGBUFSZ 数据库配置参数的大小。

相关参考:

- 『logbufsz – 日志缓冲区大小配置参数』（*《性能指南》*）
- 第 282 页的『num_log_read_io – 日志读取数』

first_active_log － 第一个活动日志文件编号

元素标识 first_active_log

元素类型 信息

表 369. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 370. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 第一个活动日志文件的文件号。

用法 将此元素与 *last_active_log* 和 *current_active_log* 元素一起使用来确定活动日志文件的范围。知道活动日志文件的范围可帮助您确定日志文件所需的磁盘空间。

您也可以使用此元素来确定哪些日志文件包含数据，以帮助您标识分割镜像支持所需的日志文件。

相关参考:

- 第 285 页的『current_active_log － 当前活动日志文件编号』
- 第 284 页的『last_active_log － 最后一个活动日志文件编号』

last_active_log － 最后一个活动日志文件编号

元素标识 last_active_log

元素类型 信息

表 371. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 372. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 最后一个活动日志文件的文件号。

用法 将此元素与 *first_active_log* 和 *current_active_log* 元素一起使用来确定活动日志文件的范围。知道活动日志文件的范围可帮助您确定日志文件所需的磁盘空间。

您也可以使用此元素来确定哪些日志文件包含数据，以帮助您标识分割镜像支持所需的日志文件。

相关参考:

- 第 285 页的『current_active_log － 当前活动日志文件编号』

• 第 284 页的『first_active_log – 第一个活动日志文件编号』

current_active_log – 当前活动日志文件编号

元素标识 current_active_log

元素类型 信息

表 373. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 374. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 DB2 数据库系统当前写入的活动日志文件的编号。

用法 将此元素与 *first_active_log* 和 *last_active_log* 元素一起使用以确定活动日志文件的范围。知道活动日志文件的范围可帮助您确定日志文件所需的磁盘空间。

您也可以使用此元素来确定哪些日志文件包含数据，以帮助您标识分割镜像支持所需的日志文件。

相关参考:

• 第 284 页的『first_active_log – 第一个活动日志文件编号』
• 第 284 页的『last_active_log – 最后一个活动日志文件编号』

current_archive_log – 当前归档日志文件编号

元素标识 current_archive_log

元素类型 信息

表 375. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 376. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

描述 DB2 数据库系统当前归档的日志文件的文件号。

如果 DB2 数据库系统未归档日志文件，则此元素的值为 SQLM_LOGFILE_NUM_UNKNOWN。

用法 使用此元素来确定归档日志文件时是否存在问题。这类问题包括:

- 归档介质速度太慢
- 归档介质不可用

数据库和应用程序活动

数据库和应用程序活动监视元素

下列各节提供有关数据库和应用程序活动的信息。
- 锁定和死锁监视元素
- 锁定等待信息监视元素
- 前滚监视监视元素
- 表空间活动监控器元素
- 表活动监控器元素
- 表重组监视元素
- SQL 游标监视元素
- SQL 语句活动监控器元素
- SQL 语句详细信息监视元素
- 子节详细信息监视元素
- 动态 SQL 监视元素
- 查询内并行性监视元素
- CPU 的使用情况监视元素
- 快照监视器监视元素
- 事件监视监视元素

锁定和死锁

锁定和死锁监视元素

下列元素提供有关锁定和死锁的信息:
- locks_held – 挂起的锁定数监视元素
- lock_list_in_use – 正在使用的总锁定列表内存监视元素
- deadlocks – 检测到的死锁监视元素
- data_partition_id – 数据分区标识监视元素data_partition_id – 数据分区标识监视元素
- lock_escals – 锁定升级数监视元素
- x_lock_escals – 互斥锁定升级数监视元素
- lock_mode – 锁定方式监视元素
- lock_status – 锁定状态监视元素
- lock_object_type – 等待的锁定对象类型监视元素
- lock_object_name – 锁定对象名监视元素
- lock_node – 锁定节点监视元素
- lock_timeouts – 锁定超时数监视元素
- locks_held_top – 挂起的最大锁定数监视元素
- dl_conns – 死锁中涉及的连接数监视元素

- lock_escalation – 锁定升级监视元素
- lock_mode_requested – 锁定的锁定方式监视元素
- deadlock_id – 死锁事件标识监视元素
- deadlock_node – 发生死锁的分区号监视元素
- participant_no – 死锁中的参与者监视元素
- participant_no_holding_lk – 挂起对应用程序所需要的对象的锁定的参与者监视元素
- rolled_back_participant_no – 回滚的应用程序参与者监视元素
- locks_in_list – 报告的锁定数监视元素
- lock_name – 锁定名称监视元素
- lock_attributes – 锁定属性监视元素
- lock_release_flags – 锁定释放标志监视元素
- lock_count – 锁定计数监视元素
- lock_hold_count – 锁定挂起计数监视元素
- lock_current_mode – 转换之前的原始锁定方式监视元素
- num_indoubt_trans – 不确定事务数监视元素num_indoubt_trans – 不确定事务数监视元素

locks_held – 挂起的锁定数

元素标识 locks_held

元素类型 标尺

表 377. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
锁	db_lock_list	基本
锁	appl_lock_list	基本

表 378. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-

描述 当前挂起的锁定数目。

用法 如果监视器信息为数据库级别，则此项表示数据库中的所有应用程序当前挂起的总锁定数。

如果监视器信息为应用程序级别，则此项表示应用程序的所有代理程序当前挂起的总锁定数。

相关参考:

- 第 289 页的『lock_escals – 锁定升级数』
- 第 290 页的『x_lock_escals – 互斥锁定升级数』
- 第 294 页的『locks_held_top – 挂起的最大锁定数』

lock_list_in_use － 使用中的锁定列表内存总量

元素标识 lock_list_in_use

元素类型 水位标记

表 379. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 使用中的锁定列表内存总量（以字节计）。

用法 此元素可以与 *locklist* 配置参数配合使用以计算锁定列表利用率。如果锁定列表利用率很高，则您可能要考虑增大该参数。有关更多信息，请参阅*管理指南*。

注: 在计算利用率时，注意下面这一点十分重要: *locklist* 配置参数是以 4K 字节页为单位分配的，而此监视元素提供的结果以字节计。

deadlocks － 检测到的死锁数

元素标识 死锁

元素类型 计数器

表 380. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	锁

可将快照监视的计数器复位。

表 381. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 发生的死锁总数。

用法 此元素可指示应用程序遇到争用问题。这些问题可能是由下列情况导致的:
- 数据库发生锁定升级
- 在系统生成的行锁定已足够的情况下应用程序显式锁定了表
- 绑定时应用程序使用了不适当的隔离级别
- 目录表已被锁定以供可重复读
- 应用程序正以不同的顺序获取相同的锁定，从而导致死锁。

可通过确定发生死锁的应用程序（或应用程序进程）来解决问题。然后可以修改应用程序以使它能够更好地并行执行。但某些应用程序可能无法并行运行。

可使用连接时间戳记监视元素（*last_reset*、*db_conn_time* 和 *appl_con_time*）来确定死锁的严重性。例如，5 分钟内发生 10 个死锁比 5 小时内发生 10 个死锁要严重得多。

以上列示的相关元素的描述还可提供其他调整建议。

相关参考:

- 第 150 页的『db_conn_time – 数据库激活时间戳记』
- 第 176 页的『appl_con_time – 连接请求启动时间戳记』
- 第 289 页的『lock_escals – 锁定升级数』
- 第 290 页的『x_lock_escals – 互斥锁定升级数』
- 第 306 页的『appl_id_holding_lk – 挂起锁定的应用程序标识』

lock_escals – 锁定升级数

元素标识 lock_escals

元素类型 计数器

表 382. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 383. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
事务	event_xact	-

描述 锁定已从若干行锁定升级至表锁定的次数。

用法 当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量，或者所有应用程序消耗的锁定列表空间达到总锁定列表空间时，锁定将会升级。可用锁定列表空间量由 *maxlocks* 和 *locklist* 配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，它将使用锁定列表中为其他应用程序分配的空间。当整个锁定列表已满时，将发生错误。

此数据项包含包括互斥锁定升级在内的所有锁定升级的计数。

以下几种原因可能会导致产生过量锁定升级:
- 锁定列表大小（*locklist*）对于并行应用程序数目而言可能太小
- 可供每个应用程序使用的锁定列表百分比（*maxlocks*）可能太小
- 一个或多个应用程序使用的锁定数可能过量。

要解决这些问题，可以:
- 增加 *locklist* 配置参数值。有关对此配置参数的描述，请参阅*管理指南*。
- 增加 *maxlocks* 配置参数值。有关对此配置参数的描述，请参阅*管理指南*。
- 标识具有大量锁定（请参阅 *locks_held_top*）的应用程序，或借助以下公式并将该值与 maxlocks 进行比较以标识在锁定列表中挂起过量锁定的应用程序:

```
(((locks held * 36) / (locklist * 4096)) * 100)
```

  。这些应用程序还可能因为在锁定列表中使用过量资源而导致其他应用程序中发生锁定升级。这些应用程序可能需要求助于使用表锁定（而不是行锁定），尽管表锁定可能导致 lock_waits 和 lock_wait_time 增加。

相关参考:

- 第 150 页的『db_conn_time – 数据库激活时间戳记』
- 第 290 页的『x_lock_escals – 互斥锁定升级数』
- 第 294 页的『locks_held_top – 挂起的最大锁定数』

x_lock_escals – 互斥锁定升级数

元素标识	x_lock_escals
元素类型	计数器

表 384. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 385. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
事务	event_xact	-

描述 锁定从若干行锁定升级至一个互斥表锁定的次数，或者针对行的互斥锁定导致表锁定成为互斥锁定的次数。

用法 其他应用程序不能访问互斥锁定挂起的数据；因此需要跟踪互斥锁定，原因是它们会影响数据的并行性。

当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量时，锁定将会升级。可用锁定列表空间量由 *locklist* 和 *maxlocks* 配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，它将使用锁定列表中为其他应用程序分配的空间。当整个锁定列表已满时，将发生错误。

有关导致出现过量互斥锁定升级的原因和解决办法，请参阅 *lock_escals*。

当共享锁定已足够时，应用程序可能会使用互斥锁定。尽管共享锁定可能不会降低锁定升级总数，但共享锁定升级可能比互斥锁定升级更好一些。

相关参考:

- 第 150 页的『db_conn_time – 数据库激活时间戳记』
- 第 289 页的『lock_escals – 锁定升级数』
- 第 176 页的『appl_con_time – 连接请求启动时间戳记』
- 第 294 页的『locks_held_top – 挂起的最大锁定数』

lock_mode – 锁定方式

元素标识	lock_mode
元素类型	信息

表 386. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁
锁	lock	锁
锁	lock_wait	锁

表 387. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 要挂起的锁定的类型。

用法 此方式可帮助您确定资源争用的源头。

根据要检查的监视器信息的类型，此元素指示下列其中一种情况:

- 另一应用程序针对此应用程序等待锁定的对象挂起的锁定类型（对于应用程序监视级别和死锁监视级别）
- 此应用程序针对对象挂起的锁定类型（对于对象锁定级别）。

此字段的值包括:

方式	锁定类型	API 常量
	没有锁定	SQLM_LNON
IS	意向共享锁定	SQLM_LOIS
IX	意向互斥锁定	SQLM_LOIX
S	共享锁定	SQLM_LOOS
SIX	与意向互斥锁定共享	SQLM_LSIX
X	互斥锁定	SQLM_LOOX
IN	无任何意向	SQLM_LOIN
Z	超级互斥锁定	SQLM_LOOZ
U	更新锁定	SQLM_LOOU
NS	下一键共享锁定	SQLM_LONS
NX	下一键互斥锁定	SQLM_LONX
W	弱互斥锁定	SQLM_LOOW
NW	下一键弱互斥锁定	SQLM_LONW

lock_status – 锁定状态

元素标识 lock_status

元素类型 信息

表 388. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁	lock	基本

表 389. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-

描述 指示锁定的内部状态。

用法 此元素可帮助说明应用程序等待获取对某个对象的锁定时所发生的情况。虽然可能已经具有对所需对象的锁定，但应用程序必须等待以获取对同一对象的另一类型的锁定。

锁定可以是下列其中一种状态:

已授予状态 指示应用程序的锁定已经处于 lock_mode 所指定的状态。

转换状态 指示应用程序尝试将拥有的锁定更改为另一类型；例如，从共享锁定更改为互斥锁定。

注: API 用户应参考包含数据库系统监视器常量定义的 *sqlmon.h* 头文件。

相关参考:

- 第 290 页的『lock_mode – 锁定方式』
- 第 292 页的『lock_object_type – 等待的锁定对象类型』
- 第 293 页的『lock_object_name – 锁定对象名称』
- 第 341 页的『table_file_id – 表文件标识』

lock_object_type – 等待的锁定对象类型

元素标识 lock_object_type

元素类型 信息

表 390. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁
锁	appl_lock_list	锁
锁	lock	基本
锁	lock_wait	锁

表 391. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 应用程序对其挂起锁定的对象的类型（对于对象锁定级别信息），或者应用程序等待对其获取锁定的对象的类型（对于应用程序级别和死锁级别信息）。

用法 此元素可帮助您确定资源争用的源头。

对象类型标识是在 sqlmon.h 中定义的。对象可能是下列其中一种类型:
- 表空间（sqlmon.h 中的 SQLM_TABLESPACE_LOCK）
- 表
- 缓冲池
- 块

- 记录（或行）
- 数据分区（sqlmon.h 中的 SQLM_TABLE_PART_LOCK）
- 内部（数据库管理器内部挂起的另一类型的锁定）
- 自动调整大小
- 自动存储器。

lock_object_name － 锁定对象名称

元素标识 lock_object_name

元素类型 信息

表 392. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁
锁	appl_lock_list	锁
锁	lock	基本

表 393. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 此元素仅供参考。它是应用程序对其挂起锁定的对象的名称（对于对象锁定级别信息），或者应用程序等待对其获取锁定的对象的名称（对于应用程序级别和死锁级别信息）。

用法 对于表级别锁定，该对象名是 SMS 和 DMS 表空间的文件标识（FID）。对于行级别锁定，该对象名是行标识（RID）。对于表空间锁定，该对象名留为空白。对于缓冲池锁定，该对象名是缓冲池的名称。

要确定挂起锁定的表，请使用 *table_name* 和 *table_schema* 而不是文件标识，原因是文件标识可能不是唯一的。

要确定挂起锁定的表空间，请使用 *tablespace_name*。

相关参考:

- 第 292 页的『lock_object_type － 等待的锁定对象类型』
- 第 312 页的『tablespace_name － 表空间名称』
- 第 333 页的『table_name － 表名称』
- 第 334 页的『table_schema － 表模式名称』

lock_node － 锁定节点

元素标识 lock_node

元素类型 信息

表 394. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句
死锁	event_dlconn	语句
带有详细信息的死锁	event_detailed_dlconn	语句

描述 锁定涉及的节点。

用法 它可以用于故障诊断。

lock_timeouts – 锁定超时次数

元素标识 lock_timeouts

元素类型 计数器

表 395. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 396. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 请求锁定对象超时（而不是授予）的次数。

用法 此元素可帮助您调整 *locktimeout* 数据库配置参数的设置。如果与正常操作级别相比锁定超时次数过多，则可能是应用程序挂起锁定的时间过长。在此情况下，此元素可能指示应该分析一些其他锁定和死锁监视元素以确定是否存在应用程序问题。

如果 *locktimeout* 数据库配置参数设置得过高，则锁定超时可能会很少。在此情况下，应用程序可能会等待很长时间来获取锁定。有关更多信息，请参阅*管理指南*。

locks_held_top – 挂起的最大锁定数

元素标识 locks_held_top

元素类型 计数器

表 397. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	-

描述 此事务期间挂起的最大锁定数。

用法 可使用此元素来确定应用程序是否达到 *maxlocks* 配置参数定义的最大可用锁定

数。此参数指示发生锁定升级之前每个应用程序可以使用的锁定列表百分比。锁定升级可能导致连接至数据库的应用程序之间的并行度下降。(有关此参数的更多信息，请参阅*管理指南*。)

因为 *maxlocks* 参数被指定为百分比并且此元素为计数器，所以可以借助以下公式进行计算以将此元素提供的计数与应用程序可挂起的总锁定数进行比较:

```
(locklist * 4096 / 36 ) * (maxlocks / 100)
```

如果您有大量锁定，则可能需要在应用程序中执行更多落实操作以便可以释放一些锁定。

相关参考:

- 第 287 页的『locks_held – 挂起的锁定数』
- 第 289 页的『lock_escals – 锁定升级数』
- 第 290 页的『x_lock_escals – 互斥锁定升级数』

dl_conns – 死锁中涉及的连接数

元素标识 dl_conns

元素类型 标尺

表 398. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-

描述 死锁中涉及的连接数。

用法 在监视应用程序中使用此元素以标识事件监视器数据流中将会出现的死锁连接事件记录数。

lock_escalation – 锁定升级

元素标识 lock_escalation

元素类型 信息

表 399. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁	lock	锁
锁	lock_wait	锁

表 400. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 指示锁定请求是否作为锁定升级的一部分。

用法 使用此元素来更好地理解死锁的原因。如果遇到涉及执行锁定升级的应用程序的死锁，则您可能想要增加锁定内存量或更改任一应用程序可请求的锁定百分比。

lock_mode_requested － 请求的锁定方式

元素标识 lock_mode_requested

元素类型 信息

表 401. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁	lock_wait	锁

表 402. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 应用程序请求的锁定方式。

用法 应用程序请求的锁定方式。此值可帮助您确定资源争用的源头。

deadlock_id － 死锁事件标识

元素标识 deadlock_id

元素类型 信息

表 403. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-
带有详细信息的死锁的历史纪录	event_detailed_dlconn	-
带有详细信息的死锁的历史纪录	event_stmt_history	-
带有详细信息的死锁的历史记录值	event_data_value	-
带有详细信息的死锁的历史记录值	event_detailed_dlconn	-
带有详细信息的死锁的历史记录值	event_stmt_history	-

描述 死锁的标识。

用法 在监视应用程序中使用此元素，以将死锁连接和语句历史记录事件记录与死锁事件记录相关联。

deadlock_node – 发生死锁的分区号

元素标识 deadlock_node

元素类型 信息

表 404. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 发生死锁的分区号。

用法 此元素仅与分区数据库有关。在监视应用程序中使用此元素，以将死锁连接事件记录与死锁事件记录相关联。

participant_no – 死锁参与者

元素标识 participant_no

元素类型 信息

表 405. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 唯一标识死锁参与者的序号。

用法 在监视应用程序中使用此元素，以将死锁连接事件记录与死锁事件记录相关联。

participant_no_holding_lk – 挂起对应用程序所需对象的锁定的参与者

元素标识 participant_no_holding_lk

元素类型 信息

表 406. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 对某个对象挂起锁定的应用程序的参与者编号，此应用程序正在等待对该对象获取锁定。

用法 此元素可帮助您确定存在资源争用的应用程序。

相关参考:

- 第 306 页的『appl_id_holding_lk – 挂起锁定的应用程序标识』

rolled_back_participant_no – 回滚的应用程序参与者

元素标识 rolled_back_participant_no

元素类型 信息

表 407. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-

描述 标识回滚的应用程序的参与者编号。

用法 系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

相关参考:

- 第 307 页的『rolled_back_appl_id – 回滚的应用程序』

locks_in_list – 报告的锁定数

元素标识 locks_in_list

元素类型 信息

表 408. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-

描述 事件监视器要报告的特定应用程序挂起的锁定数。

lock_name – 锁定名称

元素标识 lock_name

元素类型 信息

表 409. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁	lock	基本
锁	lock_wait	lock_wait

表 410. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-

描述 内部二进制锁定名称。此元素将充当锁定的唯一标识。

lock_attributes – 锁定属性

元素标识 lock_attributes

元素类型 信息

表 411. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁	lock	基本
锁	lock_wait	基本

表 412. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-

描述 锁定属性。

用法 以下是可能的锁定属性设置。每个锁定属性设置都以 sqlmon.h 中定义的位标志值为基础。

API 常量	描述
SQLM_LOCKATTR_WAIT_FOR_AVAIL	等待可用性。
SQLM_LOCKATTR_ESCALATED	由升级获取。
SQLM_LOCKATTR_RR_IN_BLOCK	块中的 RR 锁定。
SQLM_LOCKATTR_INSERT	插入锁定。
SQLM_LOCKATTR_DELETE_IN_BLOCK	块中的已删除行。
SQLM_LOCKATTR_RR	由 RR 扫描锁定。
SQLM_LOCKATTR__UPDATE_DELETE	更新／删除行锁定。
SQLM_LOCKATTR_ALLOW_NEW	允许新锁定请求。
SQLM_LOCKATTR_NEW_REQUEST	新锁定请求者。

lock_release_flags － 锁定释放标志

元素标识 lock_release_flags

元素类型 信息

表 413. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁	lock	基本
锁	lock_wait	基本

表 414. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-

描述 锁定释放标志。

用法 以下是可能的释放标志设置。每个释放标志都以 sqlmon.h 中定义的位标志值为基础。

API 常量	描述
SQLM_LOCKRELFLAGS_SQLCOMPILER	SQL 编译器进行的锁定。
SQLM_LOCKRELFLAGS_UNTRACKED	非唯一的未记录锁定。

注: 所有非指定位将用于应用程序游标。

lock_count – 锁定计数

元素标识 lock_count

元素类型 标尺

表 415. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁	lock	基本

表 416. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-

描述 要挂起的锁定上的锁定数目。

用法 此值的范围在 1 到 255 之间。获得新锁定时，此值递增，释放锁定时，此值递减。

lock_count 的值为 255 时，指示将挂起 *transaction duration lock*。此时，获得或释放锁定时，lock_count 不再递增或递减。在下列任一可能方式下，lock_count 元素的值设置为 255:

1. lock_count 因为获取新锁定而递增 255 次。
2. 显式获得事务持续时间锁定。例如使用 LOCK TABLE 语句或 INSERT 语句。

lock_hold_count – 锁定挂起计数

元素标识 lock_hold_count

元素类型 标尺

表 417. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁	lock	基本

表 418. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-

描述 放置在锁定上的挂起数目。使用 WITH HOLD 子句和一些 DB2 实用程序注册的游标放置在锁定上的挂起数。落实事务时，不会释放带有挂起的锁定。

lock_current_mode － 转换前的原始锁定方式

元素标识 lock_current_mode

元素类型 信息

表 419. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁	lock	基本
锁	lock_wait	基本

表 420. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-

描述 在锁定转换操作期间，完成转换前挂起的锁定的类型。以下是描述锁定转换的方案的示例：在更新或删除操作期间，可以等待针对目标行的 X 锁定。如果事务要挂起针对行的 S 或 V 锁定，则需要转换。此时将对 lock_current_mode 元素指定值 S 或 V，而锁定等待转换为 X 锁定。

相关概念:

- 『锁定转换』（*《性能指南》*）

num_indoubt_trans － 不确定事务数

元素标识 num_indoubt_trans

元素类型 标尺

表 421. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

描述 数据库中的不确定事务的数目。

用法 不确定事务具有未落实事务的日志空间，这可能导致日志变满。日志变满时，就不能完成更多事务。此问题的解决办法涉及以启发方式解决不确定事务的手工过程。此监视元素提供当前未完成并且必须以启发方式解决的不确定事务的数目。

锁定等待信息

锁定等待信息监视元素

下列元素提供 DB2 代理程序代表等待获取锁定的应用程序工作时返回的信息:

- lock_waits － 锁定等待监视元素
- lock_wait_time － 等待锁定的时间监视元素
- locks_waiting － 等待锁定监视器的当前代理程序数监视元素
- uow_lock_wait_time － 工作单元等待锁定的总时间监视元素

- lock_wait_start_time – 锁定等待开始时间戳记监视元素
- lock_timeout_val – 锁定超时监视元素lock_timeout_val – 锁定超时监视元素
- agent_id_holding_lock – 挂起锁定的代理程序标识监视元素
- appl_id_holding_lk – 挂起锁定的应用程序标识监视元素
- sequence_no_holding_lk – 挂起锁定的序号监视元素
- rolled_back_appl_id – 回滚的应用程序监视元素
- rolled_back_agent_id – 回滚的代理程序监视元素
- rolled_back_sequence_no – 回滚的序号监视元素
- data_partition_id – 数据分区标识监视元素

lock_waits – 锁定等待数

元素标识 lock_waits

元素类型 计数器

表 422. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 423. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 应用程序或连接等待锁定的总次数。

用法 在数据库级别，此项表示应用程序在数据库中等待锁定的总次数。

在应用程序连接级别，此项表示此连接请求锁定但必须等待（因为另一连接已经挂起针对该数据的锁定）的总次数。

此元素可与 *lock_wait_time* 配合使用以在数据库级别计算锁定的平均等待时间。此计算可在数据库级别或应用程序连接级别完成。

如果平均锁定等待时间很长，则应查找挂起许多锁定或具有锁定升级的应用程序，并把重点放在调整应用程序以改进并行性上（如果适当）。如果平均锁定等待时间很长是升级导致的，则 *locklist* 和／或 *maxlocks* 配置参数的值可能太低了。

相关概念:

- 『锁定与并行性控制』（*《性能指南》*）

相关参考:

- 第 176 页的『appl_con_time – 连接请求启动时间戳记』
- 第 303 页的『lock_wait_time – 等待锁定的时间』

lock_wait_time – 等待锁定的时间

元素标识 lock_wait_time

元素类型 计数器

表 424. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	锁
应用程序	appl	锁
锁	appl_lock_list	appl_lock_list

可将快照监视的计数器复位。

表 425. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
事务	event_xact	-

描述 等待锁定的总耗用时间。耗用时间以毫秒计。

用法 在数据库级别，此项表示所有应用程序在此数据库中等待锁定的总次数。

在应用程序连接和事务级别，此项表示此连接或事务等待授予锁定的总耗用时间。

此元素的值不包括当前仍处于锁定等待状态的代理程序的锁定等待时间。它仅包括已完成锁定等待的代理程序的锁定等待时间。

此元素可与 *lock_waits* 监视元素一起使用以计算锁定的平均等待时间。此计算可在数据库级别或应用程序连接级别执行。

使用提供耗用时间的监视元素时，应考虑:

- 耗用时间受系统负载影响，所以运行的进程越多，此耗用时间值越高。
- 要在数据库级别计算此元素，数据库系统监视器会将应用程序级别时间求和。这会导致对数据库级别的耗用时间双倍计数，原因是多个应用程序进程可能同时运行。

  为提供有意义的数据，可按上述计算锁定的平均等待时间。

相关参考:

- 第 303 页的『locks_waiting – 等待锁定的当前代理程序数』
- 第 302 页的『lock_waits – 锁定等待数』

locks_waiting – 等待锁定的当前代理程序数

元素标识 locks_waiting

元素类型 标尺

表 426. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
锁	db_lock_list	基本

描述 指示等待锁定的代理程序数。

用法 与 *appls_cur_cons* 一起使用时，此元素指示等待锁定的应用程序百分比。如果此数字很高，则表示应用程序可能具有并行性问题，您应该标识长时间挂起锁定或互斥锁定的应用程序。

相关参考:

- 第 187 页的『appls_cur_cons – 当前连接的应用程序数』

uow_lock_wait_time – 工作单元等待锁定的总时间

元素标识 uow_lock_wait_time

元素类型 计数器

表 427. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元

描述 此工作单元等待锁定时耗用的总时间。

用法 此元素可帮助您确定资源争用问题的严重性。

lock_wait_start_time – 锁定等待开始时间戳记

元素标识 lock_wait_start_time

元素类型 时间戳记

表 428. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定，时间戳记
锁	lock_wait	锁定，时间戳记

表 429. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	时间戳记
带有详细信息的死锁	event_detailed_dlconn	时间戳记

描述 此应用程序开始等待获取锁定的日期和时间，该锁定针对当前被另一应用程序锁定的对象。

用法 此元素可帮助您确定资源争用的严重性。

相关参考:

- 第 305 页的『agent_id_holding_lock – 挂起锁定的代理程序标识』

lock_timeout_val – 锁定超时

元素标识 lock_timeout_val

元素类型 信息

表 430. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
应用程序	agent	基本

描述 指示应用程序发出 SET CURRENT LOCK TIMEOUT 语句时的超时值（以秒计）。如果未执行语句，则将显示数据库级别锁定超时。

用法 可使用 SET CURRENT LOCK TIMEOUT 语句来指定应用程序代理程序等待表或索引锁定的最长持续时间。

如果应用程序等待锁定的时间过长，则可以检查 *lock_timeout_val* 值以了解该值在应用程序中是否设置得过高。如果符合应用程序逻辑的话，可修改应用程序来降低 *lock_timeout_val* 的值以允许应用程序超时。可使用 SET CURRENT LOCK TIMEOUT 语句来完成此修改。

如果应用程序经常超时，则可以检查 *lock_timeout_val* 值是否设置得过低，并在适当时提高该值。

agent_id_holding_lock – 挂起锁定的代理程序标识

元素标识 agent_id_holding_lock

元素类型 信息

表 431. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁
锁	appl_lock_list	锁
锁	lock_wait	锁

描述 代理程序的应用程序句柄，该代理程序挂起此应用程序正在等待的锁定。必须将锁定监视器组设置为“ON”以获取此信息。

用法 此元素可帮助您确定存在资源争用的应用程序。

如果此元素为 0（零）并且应用程序正在等待锁定，则这指示锁定被不确定事务所挂起。可使用 appl_id_holding_lk 或 命令行处理器 LIST INDOUBT TRANSACTIONS 命令（当事务变得不确定时，它将显示处理该事务的 CICS® 代理程序的应用程序标识）来确定不确定事务，然后对其执行落实或回滚操作。

注意，多个应用程序可挂起针对此应用程序正在等待的对象的共享锁定。有关该应用程序挂起的锁定类型的信息，请参阅 lock_mode。如果正在获取应用程序快照，则将只返回对该对象挂起锁定的其中一个代理程序标识。如果正在获取锁定快照，则将标识对该对象挂起锁定的所有代理程序标识。

相关参考:

- 第 304 页的『lock_wait_start_time － 锁定等待开始时间戳记』
- 第 306 页的『appl_id_holding_lk － 挂起锁定的应用程序标识』

appl_id_holding_lk － 挂起锁定的应用程序标识

元素标识 appl_id_holding_lk

元素类型 信息

表 432. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁
锁	appl_lock_list	锁
锁	lock_wait	锁

表 433. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 对某个对象挂起锁定的应用程序的标识，此应用程序正在等待对该对象获取锁定。

用法 此元素可帮助您确定存在资源争用的应用程序。具体地说，它可以帮助您标识挂起锁定的应用程序句柄（代理程序标识）和表标识。注意，可使用 LIST APPLICATIONS 命令来获取有关带有代理程序标识的应用程序标识的信息。但是，最好在获取快照时收集此类型的信息，原因是应用程序在运行 LIST APPLICATIONS 命令之前结束时此项将变得不可用。

注意，多个应用程序可挂起某个对象的共享锁定，此应用程序正在等待对该对象获取锁定。有关该应用程序挂起的锁定类型的信息，请参阅 lock_mode。如果正在获取应用程序快照，则将只返回对该对象挂起锁定的其中一个应用程序标识。如果正在获取锁定快照，则将返回对该对象挂起锁定的所有应用程序标识。

相关参考:

- 第 305 页的『agent_id_holding_lock － 挂起锁定的代理程序标识』
- 第 288 页的『deadlocks － 检测到的死锁数』

sequence_no_holding_lk － 挂起锁定的序号

元素标识 sequence_no_holding_lk

元素类型 信息

表 434. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
锁	appl_lock_list	基本

表 435. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 对某个对象挂起锁定的应用程序的序号，此应用程序正在等待对该对象获取锁定。

用法 此标识与 appl_id 配合使用以唯一标识对某个对象挂起锁定的事务，此应用程序正在等待对该对象获取锁定。

rolled_back_appl_id – 回滚的应用程序

元素标识 rolled_back_appl_id

元素类型 信息

表 436. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-

描述 发生死锁时回滚的应用程序标识。

用法 系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

相关参考:

- 第 191 页的『coord_agents_top – 最大协调代理程序数』

rolled_back_agent_id – 回滚的代理程序

元素标识 rolled_back_agent_id

元素类型 信息

表 437. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-

描述 发生死锁时回滚的代理程序。

用法 系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

相关参考:

- 第 191 页的『coord_agents_top – 最大协调代理程序数』

rolled_back_sequence_no – 回滚的序号

元素标识 rolled_back_sequence_no

元素类型 信息

表 438. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-

描述 发生死锁时回滚的应用程序的序号。

用法 系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

前滚监视

前滚监视监视元素

恢复数据库更改可能要花一段时间。可使用数据库系统监视器来监视恢复进度。下列元素提供有关前滚状态的信息:

- rf_timestamp – 前滚时间戳记监视元素
- ts_name – 正在前滚的表空间监视元素
- rf_type – 前滚类型监视元素
- rf_log_num – 正在前滚的日志监视元素
- rf_status – 日志阶段监视元素

运行时回滚进程的进度监视

运行时回滚的进度监视提供使用应用程序快照的回滚事件的进度信息。回滚事件包括两种类型:

工作单元回滚
包括整个事务的显式（用户调用）和隐式（强制）回滚。

保存点回滚
包括语句和应用程序级别保存点。嵌套保存点被视为单个单元，使用最外部的保存点。

提供的信息包括回滚事件的启动时间、要完成的工作总量和完成的工作。工作度量为字节。

总工作单元是需要对事务或保存点回滚的日志流的范围。

完成的工作单元显示已回滚的日志流中的相对位置。

在处理每个日志记录之后对已完成的工作进行更新。因为日志记录的大小有所不同，所以更新并非均匀执行。

GET SNAPSHOT FOR ALL APPLICATIONS 命令中的样本输出:

```
                         应用程序快照

应用程序句柄                = 6
应用程序状态                = 回滚活动
      开始时间              = 02/20/2004 12:49:27.713720
      完成工作量            = 1024000 字节
      总工作量              = 4084000 字节

             应用程序快照
```

```
应用程序句柄                 = 10
应用程序状态                 = 回滚到保存点
      开始时间               = 02/20/2004 12:49:32.832410
      完成工作量             = 102400 字节
      总工作量               = 2048000 字节
```

注: 如果回滚在快照期间不活动，则不会显示回滚元素。

rf_timestamp － 前滚时间戳记

元素标识 rf_timestamp

元素类型 时间戳记

表 439. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	时间戳记

描述 上次落实的事务的时间戳记。

用法 如果正在进行前滚，则这是前滚恢复操作所处理的上次落实事务的时间戳记。这是前滚操作的进度指示符。

相关参考:

- 第 309 页的『ts_name － 正在前滚的表空间』

ts_name － 正在前滚的表空间

元素标识 ts_name

元素类型 信息

表 440. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

描述 目前已前滚的表空间的名称。

用法 如果正在进行前滚，则此元素标识前滚涉及的表空间。

相关参考:

- 第 309 页的『rf_timestamp － 前滚时间戳记』

rf_type － 前滚类型

元素标识 rf_type

元素类型 信息

表 441. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

描述 正在进行的前滚的类型。

用法 指示是在数据库级别还是表空间级别进行恢复的指示符。

rf_log_num – 正在前滚的日志

元素标识	rf_log_num
元素类型	信息

表 442. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

描述 要处理的日志。

用法 如果正在进行前滚，则此元素标识前滚涉及的日志。

rf_status – 日志阶段

元素标识	rf_status
元素类型	信息

表 443. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

描述 恢复的状态。

用法 此元素指示恢复的进度。它指示恢复是处于撤销（回滚）阶段还是处于重做（前滚）阶段。

表空间活动

表空间活动监视元素

下列元素提供有关表空间的信息:

- tablespace_id – 表空间标识监视元素
- tablespace_name – 表空间名监视元素
- tablespace_type – 表空间类型监视元素
- tablespace_content_type – 表空间内容类型监视元素
- tablespace_state – 表空间状态监视元素
- tablespace_page_size – 表空间页大小监视元素
- tablespace_extent_size – 表空间扩展数据块大小监视元素
- tablespace_prefetch_size – 表空间预取大小监视元素
- tablespace_cur_pool_id – 当前正在使用的缓冲池监视元素
- tablespace_next_pool_id – 将在下次启动时使用的缓冲池监视元素
- tablespace_total_pages – 表空间中的总页数监视元素
- tablespace_usable_pages – 表空间中的可用页监视元素
- tablespace_used_pages – 表空间中已使用的页监视元素
- tablespace_free_pages – 表空间中的可用页监视元素
- tablespace_pending_free_pages – 表空间中的暂挂可用页监视元素

- tablespace_page_top – 表空间高水位标记监视元素
- tablespace_current_size – 当前表空间大小监视元素tablespace_current_size – 当前表空间大小监视元素
- tablespace_initial_size – 初始表空间大小监视元素tablespace_initial_size – 初始表空间大小监视元素
- tablespace_max_size – 最大表空间大小监视元素tablespace_max_size – 最大表空间大小监视元素
- tablespace_increase_size – 按字节增大监视元素tablespace_increase_size – 按字节增大监视元素
- tablespace_increase_size_percent – 按百分比增大监视元素tablespace_increase_size_percent – 按百分比增大监视元素
- tablespace_last_resize_time – 上次成功调整大小的时间监视元素tablespace_last_resize_time – 上次成功调整大小的时间监视元素
- tablespace_rebalancer_mode – 重新平衡程序方式监视元素
- tablespace_rebalancer_start_time – 重新平衡程序开始时间监视元素
- tablespace_rebalancer_restart_time – 重新平衡程序的重新启动时间监视元素
- tablespace_using_auto_storage – 使用自动存储器监视元素tablespace_using_auto_storage – 使用自动存储器监视元素
- tablespace_auto_resize_enabled – 启用自动调整大小监视元素tablespace_auto_resize_enabled – 启用自动调整大小监视元素
- tablespace_last_resize_failed – 上次失败的调整大小尝试监视元素tablespace_last_resize_failed – 上次失败的调整大小尝试监视元素
- tablespace_rebalancer_extents_remaining – 将由重新平衡程序处理的扩展数据块总数监视元素
- tablespace_rebalancer_extents_processed – 重新平衡程序已处理的扩展数据块数监视元素
- tablespace_rebalancer_last_extent_moved – 重新平衡程序移动的最后一个扩展数据块监视元素
- tablespace_rebalancer_priority – 当前的重新平衡程序优先级监视元素
- tablespace_num_quiescers – 停顿者的数目监视元素
- fs_caching – 文件系统高速缓存监视元素fs_caching – 文件系统高速缓存监视元素
- 表空间停顿者活动监控器元素
- tablespace_state_change_object_id – 状态更改对象标识监视元素
- tablespace_state_change_ts_id – 状态更改表空间标识监视元素
- tablespace_min_recovery_time – 前滚的最少恢复时间监视元素
- tablespace_num_containers – 表空间中的容器数监视元素
- 容器状态监视元素
- tablespace_num_ranges – 表空间映射中的范围数监视元素
- 表空间范围状态监视元素

tablespace_id – 表空间标识

元素标识	tablespace_id

元素类型 信息

表 444. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本
表	table	基本

表 445. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

描述 唯一表示当前数据库使用的表空间的整数。

用法 此元素的值与视图 SYSCAT.TABLESPACES 的列 TBSPACEID 中的值相匹配。

tablespace_name － 表空间名称

元素标识 tablespace_name

元素类型 信息

表 446. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本
锁	appl_lock_list	基本
锁	lock	锁
锁	lock_wait	锁

表 447. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-
表空间	tablespace_list	-

描述 表空间的名称。

用法 此元素可帮助您确定资源争用的源头。

它相当于数据库目录表 SYSCAT.TABLESPACES 中的 TBSPACE 列。在应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表空间的名称。另一个应用程序当前挂起针对此表空间的锁定。

在锁定级别，此项是应用程序当前对其挂起锁定的表空间的名称。

在表空间级别（缓冲池监视器组设置为“ON”时），此项是与所返回信息相对应的表空间的名称。

对于针对分区表挂起的表锁定，将不会返回此元素。

相关参考:

• 第 292 页的『lock_object_type － 等待的锁定对象类型』

tablespace_type － 表空间类型

元素标识 tablespace_type

元素类型 信息

表 448. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

描述 表空间的类型。

用法 此元素显示此表空间是数据库管理的表空间（DMS）还是系统管理的表空间（SMS）。

tablespace_type（它是在 sqlmon.h 中定义的）的值如下所示:

- 对于 DMS: SQLM_TABLESPACE_TYP_DMS
- 对于 SMS: SQLM_TABLESPACE_TYP_SMS

tablespace_content_type － 表空间内容类型

元素标识 tablespace_content_type

元素类型 信息

表 449. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

描述 表空间中的内容类型。

用法 表空间中的内容类型（在 sqlmon.h 中定义）可以是下列其中一项:

- 所有类型的永久数据。
  - 常规表空间: SQLM_TABLESPACE_CONTENT_ANY
  - 大型表空间: SQLM_TABLESPACE_CONTENT_LARGE
- 系统临时数据: SQLM_TABLESPACE_CONTENT_SYSTEMP
- 用户临时数据: SQLM_TABLESPACE_CONTENT_USRTEMP

tablespace_state － 表空间状态

元素标识 tablespace_state

元素类型 信息

表 450. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 此元素描述表空间的当前状态。

用法 此元素包含指示当前表空间状态的十六进制值。表空间的外部可视状态由特定状态值的十六进制和组成。例如，如果状态为“停顿: EXCLUSIVE”和“装入暂挂”，则值为 0x0004 + 0x0008，即 0x000c。db2tbst（获取表空间状态）

可用来获取与给定十六进制值相关联的表空间状态。

表 451. sqlutil.h 中列示的位定义

十六进制值	十进制值	状态
0x0	0	正常（请参阅 sqlutil.h 中的定义 SQLB_NORMAL）
0x1	1	停顿：SHARE
0x2	2	停顿：UPDATE
0x4	4	停顿：EXCLUSIVE
0x8	8	装入暂挂
0x10	16	删除暂挂
0x20	32	备份暂挂
0x40	64	正在前滚
0x80	128	前滚暂挂
0x100	256	复原暂挂
0x100	256	恢复暂挂（未使用）
0x200	512	禁用暂挂
0x400	1024	正在重组
0x800	2048	正在备份
0x1000	4096	必须定义存储器
0x2000	8192	正在复原
0x4000	16384	脱机并且不可访问
0x8000	32768	删除暂挂
0x2000000	33554432	可以定义存储器
0x4000000	67108864	存储器定义处于“最终”状态
0x8000000	134217728	在前滚之前已更改存储器定义
0x10000000	268435456	DMS 重新平衡程序处于活动状态
0x20000000	536870912	正在进行 TBS 删除
0x40000000	1073741824	正在进行 TBS 创建

tablespace_page_size － 表空间的页大小

元素标识 tablespace_page_size

元素类型 信息

表 452. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

描述 表空间使用的页大小（以字节计）。

tablespace_extent_size － 表空间的扩展数据块大小

元素标识 tablespace_extent_size

元素类型 信息

表 453. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

描述 表空间使用的扩展数据块大小。

tablespace_prefetch_size － 表空间的预取大小

元素标识 tablespace_prefetch_size

元素类型 信息

表 454. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本
表空间	tablespace_nodeinfo	基本

描述 预取程序一次从磁盘获取的最大页数。

如果启用了自动预取大小，则此元素在*表空间*逻辑数据分组中将显示值“-1”，而在 *tablespace_nodeinfo* 逻辑数据分组中显示实际值。

如果未启用自动预取大小，则此元素将在*表空间*逻辑数据分组中显示实际值，并且该元素不会出现在 *tablespace_nodeinfo* 逻辑数据分组中。

tablespace_cur_pool_id － 当前使用的缓冲池

元素标识 tablespace_cur_pool_id

元素类型 信息

表 455. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

描述 表空间当前使用的缓冲池的标识。

用法 每个缓冲池由唯一的整数标识。此元素的值与视图 SYSCAT.BUFFERPOOLS 的列 BUFFERPOOLID 中的值相匹配。

tablespace_next_pool_id － 将在下次启动时使用的缓冲池

元素标识 tablespace_next_pool_id

元素类型 信息

表 456. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

描述 表空间在下一次数据库启动时使用的缓冲池的标识。

用法 每个缓冲池由唯一的整数标识。此元素的值与视图 SYSCAT.BUFFERPOOLS 的列 BUFFERPOOLID 中的值相匹配。

tablespace_total_pages － 表空间中的总页数

元素标识 tablespace_total_pages

元素类型 信息

表 457. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本（DMS 表空间） 缓冲池（SMS 表空间）

描述 表空间中的总页数。

用法 表空间占用的总操作系统空间。对于 DMS，此项是容器大小的总和（包括开销）。对于 SMS，此项是用于此表空间中存储的表的所有文件的总和（并且仅在缓冲池开关设置为 ON 时收集此项）。

tablespace_usable_pages － 表空间中的可用页数

元素标识 tablespace_usable_pages

元素类型 信息

表 458. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本（DMS 表空间） 缓冲池（SMS 表空间）

描述 表空间中的总页数减去开销页数。

用法 此元素仅适用于 DMS 表空间。对于 SMS，此元素的值与 tablespace_total_pages 的值相同。

在表空间重新平衡期间，可用页数将包括新添加的容器的页数，但在重新平衡完成之前，这些新页可能不会反映在空闲页数中。如果未进行表空间重新平衡，则已使用页数加上空闲页数再加上暂挂空闲页数，将等于可用页数。

tablespace_used_pages － 表空间中的已使用页数

元素标识 tablespace_used_pages

元素类型 信息

表 459. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本（DMS 表空间） 缓冲池（SMS 表空间）

描述 当前在表空间中使用的（非空闲）总页数。

用法 这是用于 DMS 表空间的总页数。对于 SMS 表空间，它等于 tablespace_total_pages。

tablespace_free_pages － 表空间中的空闲页数

元素标识 tablespace_free_pages

元素类型 信息

表 460. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 表空间中当前空闲的总页数。

用法 此项仅适用于 DMS 表空间。

tablespace_pending_free_pages － 表空间中的暂挂可用页数

元素标识 tablespace_pending_free_pages

元素类型 信息

表 461. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 如果已落实或回滚所有暂挂事务，并且已经为对象请求了新的空间，则在表空间中将变得可用的页的数目。

用法 此项仅适用于 DMS 表空间。

tablespace_page_top － 表空间高水位标记

元素标识 tablespace_page_top

元素类型 信息

表 462. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 表空间中包含高水位标记的页。

用法 对于 DMS，此元素表示上一次分配的表空间扩展数据块之后第一个可用扩展数据块的页号。注意，这并非实际意义上的“高水位标记”，而是当前“水位标记”，这是因为该值可能下降。对于 SMS，此项不适用。

tablespace_using_auto_storage － 使用自动存储器

元素标识 tablespace_using_auto_storage

元素类型 信息

表 463. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

描述 此元素描述是否将表空间创建为自动存储器表空间。值为 1 意味着肯定；0 意味着否定。

用法 可使用此元素来确定是否使用自动存储器创建指定表空间（即使用 MANAGED BY AUTOMATIC STORAGE 子句创建），而不是使用显式提供的容器创建。表空间的某些容器可以在与数据库相关联的某些或所有存储器路径中。

tablespace_auto_resize_enabled － 能够自动调整大小

元素标识 tablespace_auto_resize_enabled

元素类型 信息

表 464. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

描述 此元素描述是否允许表空间自动调整大小。值为 1 意味着肯定；0 意味着否定。

用法 此元素仅适用于 DMS 表空间和非临时自动存储器表空间。如果此元素设置为 1，则能够自动调整大小。有关表空间的增长率和最大大小，请参阅 tablespace_increase_size、tablespace_increase_size_percent 和 tablespace_max_size。

tablespace_initial_size － 初始表空间大小

元素标识 tablespace_initial_size

元素类型 信息

表 465. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 自动存储器表空间的初始大小（以字节计）。

用法 对于非临时自动存储器表空间，此监视元素表示创建表空间时的初始大小（以字节计）。

tablespace_current_size － 当前表空间大小

元素标识 tablespace_current_size

元素类型 信息

表 466. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 此元素显示表空间的当前大小（以字节计）。

用法 对于 DMS 和自动存储器表空间，此元素表示所有表空间容器的总大小（以字节计）。此值等于表空间的总页数（tablespace_total_pages）乘以表空间的页大小（tablespace_page_size）的积。此元素不适用于 SMS 表空间或临时自动存储器表空间。

在为自动存储器表空间创建表空间时，当前大小可能与初始大小不匹配。当前大小的值应在页大小乘以扩展数据块大小乘以创建时初始大小的存储器路径数的积的范围内 （通常大于该积，但有时小于该积）。它将总是小于或等于 tablespace_max_size（如果设置了的话）。这是因为容器只能以完整的扩展数据块大小增长，并且必须以集合的方式增长。

tablespace_max_size － 最大表空间大小

元素标识 tablespace_max_size

元素类型 信息

表 467. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 此元素显示表空间可自动调整大小或增长至的最大大小（以字节计）。

用法 此项表示可自动调整大小的表空间可自动增长至的最大大小（以字节计）。如果此值等于 tablespace_current_size 元素，则没有空间来容纳表空间的增长。如果此元素的值为 -1，则最大大小被认为是『无限』并且表空间可自动调整大小直到文件系统已满或达到表空间的体系结构大小限制。（此限制将在*SQL Reference*中的 SQL 限制附录中描述）。此元素仅适用于能够自动调整大小的表空间。

tablespace_increase_size － 增加字节大小

元素标识 tablespace_increase_size

元素类型 信息

表 468. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 此元素显示表空间已满并且需要更多空间时自动调整大小的表空间将增加的大小（以字节计）。

用法 此项表示将添加表空间的空间量，当该表空间变满并且请求更多空间，同时又未达到最大表空间大小时，该表空间可自动调整大小。如果此元素的值为 -1（或者在快照输出中为 『AUTOMATIC』），则 DB2 将自动确定需要添加空间时的值。此元素仅适用于能够自动调整大小的表空间。

tablespace_increase_size_percent － 增加大小（以百分比计）

元素标识 tablespace_increase_size_percent

元素类型 信息

表 469. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 此元素显示表空间已满并且需要更多空间时自动调整大小的表空间将增加的量。实际字节数是根据表空间当时的大小来调整大小时确定的。

用法 此项表示将添加表空间的空间量，当该表空间变满并且请求更多空间，同时又未达到最大表空间大小时，该表空间可自动调整大小。增长率以调整表空间大小时当前表空间大小（tablespace_current_size）所占的百分比为基础。此元素仅适用于能够自动调整大小的表空间。

tablespace_last_resize_time – 上次成功调整大小的时间

元素标识 tablespace_last_resize_time

元素类型 信息

表 470. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 此元素显示用来表示上次成功增加表空间的大小的时间戳记。

用法 对于可自动调整大小的表空间，此元素表示上次表空间变满并且请求更多空间，同时又未达到最大表空间大小时，自动对该表空间添加空间的时间。此元素仅适用于能够自动调整大小的表空间。

tablespace_last_resize_failed – 上一次调整大小尝试失败

元素标识 tablespace_last_resize_failed

元素类型 信息

表 471. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 此元素描述上一次尝试自动增加表空间大小是否失败。值为 1 意味着肯定；0 意味着否定。

用法 对于自动存储器表空间，此元素可能显示任何数据库存储器路径上都没有留下空间。对于非自动存储器表空间，失败意味着因为文件系统已满而未能扩展其中一个容器。失败的另一个原因是已达到表空间的最大大小。此元素仅适用于能够自动调整大小的表空间。

tablespace_rebalancer_mode – 重新平衡程序方式

元素标识 tablespace_rebalancer_mode

元素类型 信息

表 472. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 表示是进行正向重新平衡还是逆向重新平衡的整数。其值（它是在 sqlmon.h 中定义的）如下所示:

- 不进行重新平衡：SQLM_TABLESPACE_NO_REBAL
- 正向：SQLM_TABLESPACE_FWD_REBAL
- 逆向：SQLM_TABLESPACE_REV_REBAL

用法 此项可用作一个指示符，用来指示当前重新平衡过程是从表空间除去空间还是向表空间添加空间。此项仅适用于 DMS 表空间。

tablespace_rebalancer_start_time － 重新平衡程序启动时间

元素标识 tablespace_rebalancer_start_time

元素类型 信息

表 473. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 表示重新平衡程序最初启动时间的时间戳记。

用法 此项用于指示重新平衡程序最初启动的时间。它可以用来派生度量，以测量运行重新平衡程序的速度和完成重新平衡的估计时间。此项仅适用于 DMS 表空间。

tablespace_rebalancer_restart_time － 重新平衡程序重新启动时间

元素标识 tablespace_rebalancer_restart_time

元素类型 信息

表 474. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 表示重新平衡程序在暂停或停止后重新启动的时间戳记。

用法 此项可用作重新平衡程序的完成级别的指示符。它将说明重新平衡程序重新启动的时间，并且允许派生重新平衡程序的速度和直到完成所耗用的时间。此项仅适用于 DMS 表空间。

tablespace_rebalancer_extents_remaining － 重新平衡程序要处理的扩展数据块总数

元素标识 tablespace_rebalancer_extents_remaining

元素类型 信息

表 475. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 要移动的扩展数据块数目。此值是在重新平衡程序启动时间或重新启动时间计算的（选择最近的时间）。

用法 此项可用作重新平衡程序的完成级别的指示符。可通过记录此元素在一段时间内的更改来监视重新平衡进度。可使用 tablespace_state 来检查重新平衡是否完成。此项仅适用于 DMS 表空间。

tablespace_rebalancer_extents_processed - 重新平衡程序已经处理的扩展数据块数

元素标识 tablespace_rebalancer_extents_processed

元素类型 信息

表 476. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 自重新平衡程序启动或重新启动后（选择最近的时间）重新平衡程序已经移动的扩展数据块数。

用法 此项可用作重新平衡程序的完成级别的指示符。可通过记录此元素在一段时间内的更改来监视重新平衡进度。可使用 tablespace_state 和 rebalance_mode 来检查重新平衡是否完成。此项仅适用于 DMS 表空间。

tablespace_rebalancer_last_extent_moved - 重新平衡程序移动的最后一个扩展数据块

元素标识 tablespace_rebalancer_last_extent_moved

元素类型 信息

表 477. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 重新平衡程序移动的最后一个扩展数据块。

用法 此项可用作重新平衡程序的完成级别的指示符。可通过记录此元素在一段时间内的更改来监视重新平衡进度。可使用 tablespace_state 和 rebalance_mode 来检查重新平衡是否完成。此项仅适用于 DMS 表空间。

tablespace_rebalancer_priority - 当前重新平衡程序优先级

元素标识 tablespace_rebalancer_priority

元素类型 信息

表 478. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 在数据库中运行的重新平衡程序的优先级。

用法 此项仅适用于 DMS 表空间。

tablespace_num_quiescers － 停顿者数目

元素标识	tablespace_num_quiescers
元素类型	信息

表 479. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 停顿表空间的用户数目（可以在范围 0 到 5 之间）。

用法 此值表示停顿表空间的代理程序数（以“SHARE”、“UPDATE”或“EXCLUSIVE”方式）。对于每个停顿者，将在 tablespace_quiescer 逻辑数据组中返回以下信息:

- 停顿者的用户授权标识
- 停顿者的代理程序标识
- 因为停顿而导致此表空间停顿的对象的表空间标识
- 因为停顿而导致此表空间停顿的对象的对象标识
- 停顿状态

表空间停顿者活动监视元素

表空间停顿者活动监视元素: 下列元素提供有关表空间停顿者活动的信息:

- quiescer_auth_id － 停顿者用户授权标识监视元素
- quiescer_agent_id － 停顿者代理程序标识监视元素
- quiescer_ts_id － 停顿者表空间标识监视元素
- quiescer_obj_id － 停顿者对象标识监视元素
- quiescer_state － 停顿者状态监视元素

quiescer_auth_id － 停顿者用户授权标识:

元素标识	quiescer_auth_id
元素类型	信息

表 480. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

描述 具有停顿状态的用户的授权标识。

用法 使用此元素来确定停顿表空间的人员。

quiescer_agent_id － 停顿者代理程序标识:

元素标识	quiescer_agent_id
元素类型	信息

表 481. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

描述 具有停顿状态的代理程序的代理程序标识。

用法 将此元素与 quiescer_auth_id 配合使用，以确定停顿表空间的人员。

quiescer_ts_id – 停顿者表空间标识:

元素标识 quiescer_ts_id

元素类型 信息

表 482. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

描述 导致表空间停顿的对象的表空间标识。

用法 将此元素与 quiescer_obj_id 和 quiescer_auth_id 配合使用，以确定停顿表空间的人员。此元素的值与视图 SYSCAT.TABLES 的列 TBSPACEID 中的值相匹配。

quiescer_obj_id – 停顿者对象标识:

元素标识 quiescer_obj_id

元素类型 信息

表 483. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

描述 导致表空间停顿的对象的对象标识。

用法 将此元素与 quiescer_ts_id 和 quiescer_auth_id 配合使用以确定停顿表空间的人员。此元素的值与视图 SYSCAT.TABLES 的列 TABLEID 中的值相匹配。

quiescer_state – 停顿者状态:

元素标识 quiescer_state

元素类型 信息

表 484. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

描述 要完成的停顿类型（如“SHARE”、“INTENT TO UPDATE”或“EXCLUSIVE”）。

用法 此元素的值与 sqlutil.h 中的常量 SQLB_QUIESCED_SHARE、SQLB_QUIESCED_UPDATE 或 SQLB_QUIESCED_EXCLUSIVE 的值相匹配。

tablespace_state_change_object_id － 状态更改对象标识

元素标识 tablespace_state_change_object_id

元素类型 信息

表 485. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 导致表空间状态设置为“装入暂挂”或“删除暂挂”的对象。

用法 仅当表空间状态为“装入暂挂”或“删除暂挂”时，此元素才有意义。如果此元素的值非零，则此元素的值与视图 SYSCAT.TABLES 的列 TABLEID 中的值相匹配。

tablespace_state_change_ts_id － 状态更改表空间标识

元素标识 tablespace_state_change_ts_id

元素类型 信息

表 486. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 如果表空间状态为“装入暂挂”或“删除暂挂”，则此项显示导致设置表空间状态的对象的表空间标识。

用法 仅当表空间状态为“装入暂挂”或“删除暂挂”时，此元素才有意义。如果此元素的值非零，则此元素的值与视图 SYSCAT.TABLES 的列 TABLESPACEID 中的值相匹配。

tablespace_min_recovery_time － 前滚的最短恢复时间

元素标识 tablespace_min_recovery_time

元素类型 信息

表 487. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 显示可前滚表空间的最早时间点的时间戳记。

用法 只有不为零时才显示。

tablespace_num_containers － 表空间中的容器数目

元素标识 tablespace_num_containers

元素类型 信息

表 488. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 表空间中的容器总数。

容器状态

容器状态监视元素: 下列元素提供有关容器状态的信息:

- container_id – 容器标识监视元素
- container_name – 容器名监视元素
- container_type – 容器类型监视元素
- container_total_pages – 容器中的总页数监视元素
- container_usable_pages – 容器中的可用页数监视元素
- container_stripe_set – 条形集监视元素
- container_accessible – 容器的可访问性监视元素

container_id – 容器标识:

元素标识 container_id

元素类型 信息

表 489. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

描述 在表空间中唯一定义容器的整数。

用法 此元素可与元素 container_name、container_type、container_total_pages、container_usable_pages、 container_stripe_set 和 container_accessible 一起使用来描述容器。

container_name – 容器名称:

元素标识 container_name

元素类型 信息

表 490. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

描述 容器的名称。

用法 此元素可与元素 container_id、container_type、container_total_pages、container_usable_pages、 container_stripe_set 和 container_accessible 一起使用来描述容器。

container_type – 容器类型:

元素标识 container_type

元素类型 信息

表 491. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

描述 容器的类型。

用法 此元素返回容器的类型，它可能是目录路径（仅适用于 SMS）、文件（适用于 DMS）或原始设备（适用于 DMS）。此元素可与元素 container_id、container_name、 container_total_pages、container_usable_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。

在 sqlutil.h 中定义的值如下所示:

- 目录路径（SMS）： SQLB_CONT_PATH
- 原始设备（DMS）： SQLB_CONT_DISK
- 文件（DMS）： SQLB_CONT_FILE
- 条带磁盘（DMS）： SQLB_CONT_STRIPED_DISK
- 条带文件（DMS）： SQLB_CONT_STRIPED_FILE

container_total_pages － *容器中的总页数:*

元素标识 container_total_pages

元素类型 信息

表 492. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本（DMS 表空间） 缓冲池（SMS 表空间）

描述 容器占用的总页数。

用法 此元素可与元素 container_id、container_name、container_type、container_usable_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。

container_usable_pages － *容器中的可用页数:*

元素标识 container_usable_pages

元素类型 信息

表 493. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本（DMS 表空间） 缓冲池（SMS 表空间）

描述 容器中总的可用页数。

用法 此元素可与元素 container_id、container_name、container_type、container_total_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。对于 SMS 表空间，此值与 container_total_pages 相同。

***container_stripe_set* – 条带集:**

元素标识	container_stripe_set
元素类型	信息

表 494. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

描述 容器所属的条带集。

用法 此元素可与元素 container_id、container_name、container_type、container_total_pages、container_usable_pages 和 container_accessible 一起使用来描述容器。此项仅适用于 DMS 表空间。

***container_accessible* – 容器的可访问性:**

元素标识	container_accessible
元素类型	信息

表 495. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

描述 此元素描述能否访问容器（1 表示可以访问，0 表示不能访问）。

用法 此元素可与元素 container_id、container_name、container_type、container_total_pages、container_usable_pages 和 container_stripe_set 一起使用来描述容器。

tablespace_num_ranges – 表空间映射中的范围数

元素标识	tablespace_num_ranges
元素类型	信息

表 496. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

描述 表空间映射中的范围（条目）数。此项的范围在 1 到 100 之间（通常小于 12）。仅 DMS 表空间存在表空间映射。

表空间范围状态

表空间范围状态监视元素: 表空间映射用于将逻辑表空间页号映射至物理磁盘位置。该映射由一系列范围组成。

例如，范围可能为如下所示:

条带	范围	最大页	最大扩展数据块	起始条带	结束条带	调整	容器数	容器
0	[0]	249	124	0	124	0	1	(0)

条带	范围	最大页	最大扩展数据块	起始条带	结束条带	调整	容器数	容器
1	[1]	999	499	125	249	0	3	(0,1,2)
2	[2]	1499	749	250	374	0	1	(1,2)

对于每个范围，将在快照中返回以下信息（后跟模板）：

- range_stripe_set_number – 条形集数监视元素
- range_number – 范围号监视元素
- range_max_page_number – 范围中的最大页监视元素
- range_max_extent – 范围中的最大扩展数据块监视元素
- range_start_stripe – 启动条形集监视元素
- range_end_stripe – 结束条形集监视元素
- range_adjustment – 范围调整监视元素
- range_num_containers – 范围中的容器数监视元素
- 容器数组（列示属于该范围的容器）： 此数组的大小由表空间中的容器总数确定。
  - range_container_id – 范围容器监视元素
  - range_offset – 范围偏移监视元素

range_stripe_set_number – 条带集编号:

元素标识 range_stripe_set_number

元素类型 信息

表 497. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 此值表示范围所在的条带集。

用法 此元素仅适用于 DMS 表空间。

range_number – 范围编号:

元素标识 range_number

元素类型 信息

表 498. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 此值表示表空间映射内的范围的编号。

用法 此元素仅适用于 DMS 表空间。

range_max_page_number – 范围中的最大页:

元素标识 range_max_page_number

元素类型 信息

表 499. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 此值表示范围映射的最大页号。

用法 此元素仅适用于 DMS 表空间。

range_max_extent – 范围中的最大扩展数据块:

元素标识 range_max_extent

元素类型 信息

表 500. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 此值表示范围映射的最大扩展数据块编号。

用法 此元素仅适用于 DMS 表空间。

range_start_stripe – 起始条带:

元素标识 range_start_stripe

元素类型 信息

表 501. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 此值表示范围中的第一个条带的编号。

用法 此元素仅适用于 DMS 表空间。

range_end_stripe – 结束条带:

元素标识 range_end_stripe

元素类型 信息

表 502. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 此值表示范围中的最后一个条带的编号。

用法 此元素仅适用于 DMS 表空间。

range_adjustment – 范围调整:

元素标识 range_adjustment

元素类型 信息

表 503. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 此值表示容器数组中范围实际开始的位移。

用法 此元素仅适用于 DMS 表空间。

range_num_containers – 范围中的容器数:

元素标识 range_num_containers

元素类型 信息

表 504. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 此值表示当前范围中的容器数目。

用法 此元素仅适用于 DMS 表空间。

range_container_id – 范围容器:

元素标识 range_container_id

元素类型 信息

表 505. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 在范围内唯一定义容器的整数。

用法 此元素仅适用于 DMS 表空间。

range_offset – 范围位移:

元素标识 range_offset

元素类型 信息

表 506. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

描述 从范围所属的条带集开头的条带 0 开始的位移。

用法 此元素仅适用于 DMS 表空间。

fs_caching – 文件系统高速缓存

元素标识 fs_caching

元素类型 信息

表 507. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

表 508. 事件监视信息

事件类型	逻辑数据分组	监视开关
表空间	event_tablespace	-

描述 指示特定表空间是否使用文件系统高速缓存。

表活动

表活动监视元素

下列元素提供有关表的信息:
- table_type – 表类型监视元素
- table_name – 表名监视元素
- table_schema – 表模式名监视元素
- rows_deleted – 删除的行数监视元素
- rows_inserted – 插入的行数监视元素
- rows_updated – 更新的行数监视元素
- rows_selected – 选择的行数监视元素
- rows_written – 写入的行数监视元素
- rows_read – 读取的行数监视元素
- overflow_accesses – 对溢出记录的访问次数监视元素
- int_rows_deleted – 删除的内部行数监视元素
- int_rows_updated – 更新的内部行数监视元素
- int_rows_inserted – 插入的内部行数监视元素
- table_file_id – 表文件标识监视元素
- page_reorgs – 页重新组织数监视元素
- data_object_pages – 数据对象页数监视元素data_object_pages – 数据对象页数监视元素
- index_object_pages – 索引对象页数监视元素index_object_pages – 索引对象页数监视元素
- lob_object_pages – LOB 对象页数监视元素lob_object_pages – LOB 对象页数监视元素
- long_object_pages – 长整型对象页数监视元素long_object_pages – 长整型对象页数监视元素
- data_partition_id – 数据分区标识监视元素data_partition_id – 数据分区标识监视元素

table_type – 表类型

元素标识 table_type

元素类型 信息

表 509. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本

表 510. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

描述 与返回的信息相对应的表的类型。

用法 可使用此元素来帮助标识与返回的信息相对应的表。如果该表是用户表或系统目录表，则可使用 *table_name* 和 *table_schema* 来标识该表。

表的类型可能是下列其中一项:

- 用户表。
- 已废弃的用户表。
- 临时表。将返回有关临时表的信息，即使这些表在使用后未保留在数据库中。您会发现有关此类型的表的信息仍然有用。
- 系统目录表。

相关参考:

- 第 341 页的『table_file_id － 表文件标识』

table_name － 表名称

元素标识 table_name

元素类型 信息

表 511. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本
应用程序	appl	锁
锁	appl_lock_list	锁
锁	lock	锁
锁	lock_wait	锁

表 512. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-
死锁	lock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 表的名称。

用法 此元素与 *table_schema* 一起使用可帮助您确定资源争用的源头。

在应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表，原因是它目前被另一应用程序锁定。对于快照监视，仅当“锁定”监视器组信息设置为“ON”并且 *lock_object_type* 指示应用程序正在等待获取表锁定时，此项才有效。

对于对象锁定级别的快照监视，将对表级别和行级别锁定返回此项。在此级别报告的表就是此应用程序对其挂起这些锁定的表。

对于表级别的快照和事件监视，此项是对其收集信息的表。对于临时表，*table_name* 的格式为 『TEMP (*n*, *m*)』，其中:

- *n* 是表空间标识
- *m* 是 *table_file_id* 元素

相关参考:

- 第 334 页的『table_schema – 表模式名称』
- 第 292 页的『lock_object_type – 等待的锁定对象类型』
- 第 293 页的『lock_object_name – 锁定对象名称』

table_schema – 表模式名称

元素标识 table_schema

元素类型 信息

表 513. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本
应用程序	appl	锁
锁	appl_lock_list	锁
锁	lock	锁
锁	lock_wait	锁

表 514. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-
死锁	lock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 表的模式。

用法 此元素与 *table_name* 一起使用可帮助您确定资源争用的源头。

对于应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表的模式，原因是它目前被另一应用程序锁定。仅当 *lock_object_type* 指示应用程序正在等待获取表锁定时，才设置此元素。对于应用程序级别和应用程序锁定级别的快照监视，仅当“锁定”监视器组信息设置为“ON”时此项才有效。

对于对象锁定级别的快照监视，将对表级别和行级别锁定返回此项。在此级别报告的表就是此应用程序对其挂起这些锁定的表。

对于表级别的快照和事件监视，此元素标识对其收集信息的表的模式。对于临时表，*table_schema* 的格式为 『*<agent_id><auth_id>*』，其中:

- *agent_id* 是创建临时表的应用程序的句柄
- *auth_id* 是应用程序用来连接至数据库的授权标识

相关参考:

- 第 333 页的『table_name － 表名称』
- 第 292 页的『lock_object_type － 等待的锁定对象类型』
- 第 293 页的『lock_object_name － 锁定对象名称』

rows_deleted － 删除的行数

元素标识 rows_deleted

元素类型 计数器

表 515. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 516. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 此项是尝试删除的行数。

用法 可使用此元素来了解数据库内的当前活动级别。

此计数未包括 *int_rows_deleted* 中的尝试计数。

相关参考:

- 第 339 页的『int_rows_deleted － 删除的内部行数』

rows_inserted － 插入的行数

元素标识 rows_inserted

元素类型 计数器

表 517. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本

表 517. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 518. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 此项是尝试插入的行数。

用法 可使用此元素来了解数据库内的当前活动级别。

在联合系统中，每个 INSERT 语句可插入多行，原因是联合服务器会在适当时将 INSERT FROM SUBSELECT 推送至数据源。

此计数未包括 *int_rows_inserted* 中的尝试计数。

rows_updated － 更新的行数

元素标识 rows_updated

元素类型 计数器

表 519. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 520. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 此项是尝试更新的行数。

用法 可使用此元素来了解数据库内的当前活动级别。

此值不包括 *int_rows_updated* 中的更新计数。但是，将对每个更新计算多个更新语句更新的行数。

相关参考:

- 第 340 页的『int_rows_updated － 更新的内部行数』

rows_selected － 选择的行数

元素标识 rows_selected

元素类型 计数器

表 521. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 522. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 此项是已选择并且返回至应用程序的行数。

用法 可使用此元素来了解数据库内的当前活动级别。

此元素不包括对 COUNT(*) 或连接这样的操作读取的行计数。

对于联合系统，可计算将数据源中的一行返回至联合服务器的平均时间:

```
平均时间 = 返回的行数 / 聚集查询响应时间
```

可使用这些结果来修改 SYSCAT.SERVERS 中的 CPU 速度或通信速度参数。修改这些参数会影响优化器是否将请求发送至数据源。

注: 如果被监视的网关为 DB2 数据库版本 7.2 或更低版本，则将在 dcs_dbase 和 dcs_appl 快照监视器逻辑数据组中收集此元素。

相关参考:

- 第 357 页的『select_sql_stmts － 执行的 SELECT SQL 语句数』

rows_written － 写入的行数

元素标识 rows_written

元素类型 计数器

表 523. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本
应用程序	appl	基本
应用程序	stmt	基本
应用程序	subsection	语句
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 524. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
表	event_table	-
语句	event_stmt	-
事务	event_xact	-

描述 此项是表中更改（插入、删除或更新）的行数。

用法 如果表级别信息的值很高，则指示该表的使用频率太高，您可能使用“运行统计”（RUNSTATS）实用程序来保持用于此表的程序包的效率。

对于应用程序连接和语句，此元素包括临时表中插入、更新和删除的行数。

在应用程序、事务和语句级别，此元素对于分析相对活动级别和标识调整候选对象会非常有用。

相关参考:

- 第 338 页的『rows_read – 读取的行数』
- 第 341 页的『int_rows_inserted – 插入的内部行数』
- 第 339 页的『int_rows_deleted – 删除的内部行数』
- 第 340 页的『int_rows_updated – 更新的内部行数』

rows_read – 读取的行数

元素标识 rows_read

元素类型 计数器

表 525. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
表	table	表
应用程序	appl	基本
应用程序	stmt	基本
应用程序	subsection	语句
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 526. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
表	event_table	-
语句	event_stmt	-
事务	event_xact	-

描述 此项是从表中读取的行数。

用法 此元素帮助标识使用频率很高并且您可能想要为其创建附加索引的表。为避免保留不必要的索引，可使用*管理指南*中描述的 SQL EXPLAIN 语句来确定程序包是否使用索引。

此计数**不是**返回至调用应用程序的行数。而是必须读取以返回结果集的行数。例如，以下语句向应用程序返回一行，但读取了许多行以确定平均薪水:

```
SELECT AVG(SALARY) FROM USERID.EMPLOYEE
```

此计数包括 *overflow_accesses* 中的值。另外，此计数不包括任何索引访问。即，如果访问方案仅使用索引访问并且该表不会查看实际行，则 rows_read 不会递增。

相关参考:

• 第 337 页的『rows_written － 写入的行数』
• 第 339 页的『overflow_accesses － 对溢出记录的访问次数』

overflow_accesses － 对溢出记录的访问次数

元素标识 overflow_accesses

元素类型 计数器

表 527. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本

可将快照监视的计数器复位。

表 528. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

描述 对此表的溢出行的访问（读取和写入）次数。

用法 溢出行指示发生了数据分段。如果此数字很高，则可以通过使用 REORG 实用程序重组该表（这会清除此分段）来改进表的性能。

如果某行已更新并且无法再装入到一开始写入的数据页中，则该行将会溢出。这种情况通常是因为更新 VARCHAR 或 ALTER TABLE 语句造成的。

相关参考:

• 第 337 页的『rows_written － 写入的行数』
• 第 338 页的『rows_read － 读取的行数』

int_rows_deleted － 删除的内部行数

元素标识 int_rows_deleted

元素类型 计数器

表 529. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 530. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-

描述 此项是因为内部活动而从数据库中删除的行数。

用法 此元素可帮助了解您不熟悉的数据库管理器的内部活动。如果此活动很多，则您可能想要评估表设计以确定对数据库定义的引用约束或触发器是否必要。

内部删除活动可能是由下列原因引起的:
- 强制实施 ON CASCADE DELETE 引用约束的级联删除
- 被触发的触发器。

相关参考:

- 第 335 页的『rows_deleted – 删除的行数』

int_rows_updated – 更新的内部行数

元素标识 int_rows_updated

元素类型 计数器

表 531. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 532. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-

描述 此项是因为内部活动而从数据库中更新的行数。

用法 此元素可帮助了解您不熟悉的数据库管理器的内部活动。如果此活动很多，则您可能想要评估表设计以确定对数据库定义的引用约束是否必要。

内部更新活动可能是由下列原因引起的:
- 强制使用 ON DELETE SET NULL 规则定义的引用约束的 *set null* 行更新
- 被触发的触发器。

相关参考:
- 第 336 页的『rows_updated – 更新的行数』

int_rows_inserted – 插入的内部行数

元素标识 int_rows_inserted

元素类型 计数器

表 533. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 534. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-

描述 因为触发器导致的内部活动而插入到数据库中的行数。

用法 此元素可帮助您了解数据库管理器的内部活动。如果此活动很多，则您可能想要评估设计以确定是否可以改变它以降低此活动。

相关参考:
- 第 335 页的『rows_inserted – 插入的行数』

table_file_id – 表文件标识

元素标识 table_file_id

元素类型 信息

表 535. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁
表	table	基本
锁	appl_lock_list	锁
锁	lock	锁

表 536. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-

描述 此项是表的文件标识（FID）。

用法 此元素仅供参考。返回它是为了获取与数据库系统监视器的先前版本的兼容性，并且它可能**未**唯一地标识该表。使用 *table_name* 和 *table_schema* 以标识该表。

相关参考:

- 第 333 页的『table_name － 表名称』
- 第 334 页的『table_schema － 表模式名称』
- 第 332 页的『table_type － 表类型』

page_reorgs － 页重组数

元素标识 page_reorgs

元素类型 计数器

表 537. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本

可将快照监视的计数器复位。

表 538. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

描述 对表执行的页重组数。

用法

尽管页具有足够空间，但可能会因为下列情况而导致页形成碎片:

- 插入新行时
- 更新现有行并且更新导致记录大小增加时

页形成碎片时可能需要重组。重组会将所有碎片空间移至连续区域，可在该区域中写入新记录。这种页重组可能需要几千个指令。它还会生成操作的日志记录。

如果页重组过多，则可能导致插入性能无法达到最优。可使用 REORG TABLE 实用程序来重组表并消除碎片。还可对 ALTER TABLE 语句使用 APPEND 参数来指示将所有插入追加在表的结尾以避免页重组。

如果行更新导致行长度增加，则页可能有足够的空间来容纳新行，但可能需要页重组来对该空间进行碎片整理。如果页没有足够的空间来容纳增大的新行，则将创建导致在读取期间产生 *overflow_accesses* 的溢出记录。可通过使用固定长度的列而不是可变长度的列来避免出现这两种情况。

相关参考:

- 第 335 页的『rows_inserted – 插入的行数』
- 第 336 页的『rows_updated – 更新的行数』

data_object_pages – 数据对象页数

元素标识 data_object_pages

元素类型 信息

表 539. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本

表 540. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

描述 表消耗的磁盘页数。此大小仅表示基本表大小。索引对象、LOB 数据和长整型数据消耗的空间分别由 *index_object_pages*、*lob_object_pages* 和 *long_object_pages* 报告。

用法 此元素提供了一种机制，可用来查看特定表消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内表的增长率。

相关参考:

- 第 343 页的『index_object_pages – 索引对象页数』
- 第 344 页的『lob_object_pages – LOB 对象页数』
- 第 344 页的『long_object_pages – 长整型对象页数』
- 第 253 页的『xda_object_pages – XDA 对象页数』

index_object_pages – 索引对象页数

元素标识 index_object_pages

元素类型 信息

表 541. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本

表 542. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

描述 对表定义的所有索引消耗的磁盘页数。

用法 此元素提供了一种机制，可用来查看对特定表定义的索引消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内索引的增长率。不会对分区表返回此元素。

lob_object_pages – LOB 对象页数

元素标识 lob_object_pages

元素类型 信息

表 543. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本

表 544. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

描述 LOB 数据消耗的磁盘页数。

用法 此元素提供了一种机制，可用来查看特定表中的 LOB 数据消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内 LOB 数据的增长率。

long_object_pages – 长整型对象页数

元素标识 long_object_pages

元素类型 信息

表 545. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table	基本

表 546. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

描述 表中的长整型数据消耗的磁盘页数。

用法 此元素提供了一种机制，可用来查看特定表中的长整型数据消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内长整型数据的增长率。

表重组

表重组监视元素

下列元素提供有关表重组的信息:

- reorg_type – 表重组属性监视元素
- reorg_status – 表重组状态监视元素
- reorg_phase – 重组阶段监视元素
- reorg_phase_start – 重组阶段开始时间监视元素
- reorg_max_phase – 最大重组阶段监视元素
- reorg_current_counter – 重组进度监视元素
- reorg_max_counter – 重组的总工作量监视元素

- reorg_completion － 重组完成标志监视元素
- reorg_start － 表重组开始时间监视元素
- reorg_end － 表重组结束时间监视元素
- reorg_index_id － 用于重新组织表的索引监视元素
- reorg_tbspc_id － 重新组织了表或数据分区的表空间监视元素
- reorg_rows_compressed － 压缩的行数监视元素reorg_rows_compressed － 压缩的行数监视元素
- reorg_rows_rejected_for_compression － 拒绝压缩的行数监视元素reorg_rows_rejected_for_compression － 拒绝压缩的行数监视元素
- data_partition_id － 数据分区标识监视元素data_partition_id － 数据分区标识监视元素

reorg_type － 表重组属性

元素标识 reorg_type

元素类型 信息

表 547. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 表重组属性设置。

用法 可能的属性设置如下所示。每个属性设置都基于 db2ApiDf.h 中定义的位标志值。

- 允许写访问：DB2REORG_ALLOW_WRITE
- 允许读访问：DB2REORG_ALLOW_READ
- 不允许访问：DB2REORG_ALLOW_NONE
- 通过索引扫描重新集群：DB2REORG_INDEXSCAN
- 重组长型字段 LOB 数据：DB2REORG_LONGLOB
- 不截断表：DB2REORG_NOTRUNCATE_ONLINE
- 替换压缩字典：DB2REORG_RESET_DICTIONARY
- 保留压缩字典：DB2REORG_KEEP_DICTIONARY

除了上述属性设置以外，在 GET SNAPSHOT FOR TABLES 命令的 CLP 输出中还列示了下列属性。这些属性设置基于其他属性设置值或表重组监视元素值。

- 重新集群：如果 reorg_index_id 监视元素值不为零，则表重组操作具有此属性。
- 重新声明：如果 reorg_index_id 监视元素值为零，则表重组操作具有此属性。
- 原位表重组：如果 reorg_status 监视元素值不为空，则表示正在使用原位（联机）重组方法。
- 表重组：如果 reorg_phase 监视元素值不为空，则表示正在使用传统（脱机）重组方法。
- 通过表扫描重新集群：如果未设置 DB2REORG_INDEXSCAN 标志，则表重组操作具有此属性。

- 仅重组数据：如果未设置 DB2REORG_LONGLOB 标志，则表重组操作具有此属性。

reorg_status － 表重组状态

元素标识 reorg_status

元素类型 信息

表 548. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 归位（联机）表或数据分区级别重组的状态。此项不适用于经典（脱机）表重组。

用法 归位表或数据分区重组可能处于下列其中一种状态（状态与它们在 sqlmon.h 中的相应定义列示在一起）：

- 启动／继续：SQLM_REORG_STARTED
- 暂停：SQLM_REORG_PAUSED
- 停止：SQLM_REORG_STOPPED
- 完成：SQLM_REORG_COMPLETED
- 截断：SQLM_REORG_TRUNCATE

reorg_phase － 重组阶段

元素标识 reorg_phase

元素类型 信息

表 549. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 指示表的重组阶段。对于分区表来说，此元素还将指示每个数据分区的重组阶段。此元素仅适用于脱机表重组。

用法 对于分区表来说，重组是逐个数据分区进行的。对于传统表重组来说，可能的阶段如下所示（这些阶段与它们在 sqlmon.h 中的相应定义一起列示）：

- 排序：SQLM_REORG_SORT
- 构建：SQLM_REORG_BUILD
- 替换：SQLM_REORG_REPLACE
- 索引重新创建：SQLM_REORG_INDEX_RECREATE
- 字典构建：SQLM_REORG_DICT_SAMPLE

对于分区表来说，“索引重新创建”阶段是对非分区索引发生的。仅当每个数据分区上的所有先前阶段成功完成后，reorg_phase 元素才会指示“索引重新创建”阶段。

reorg_phase_start － 重组阶段开始时间

元素标识 reorg_phase_start

元素类型 时间戳记

表 550. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 表重组阶段的开始时间。对于分区表来说，此元素还将指示每个数据分区的重组阶段的开始时间。在索引重建阶段，所有数据分区的数据组将同时更新。

reorg_max_phase － 最大重组阶段

元素标识 reorg_max_phase

元素类型 信息

表 551. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 重组处理期间将发生的最大重组阶段数。此项仅适用于经典（脱机）重组。值的范围为 2 到 4（[SORT]，BUILD，REPLACE,[INDEX_RECREATE]）。

reorg_current_counter － 重组进度

元素标识 reorg_current_counter

元素类型 信息

表 552. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 指示重组完成量的进度单元。此值表示的进度与 reorg_max_counter 的值相关联，它表示要完成的表重组的总量。可使用以下公式来确定已完成的表重组的百分比:

```
表重组进度 = reorg_current_counter / reorg_max_counter * 100
```

reorg_max_counter － 重组总量

元素标识 reorg_max_counter

元素类型 信息

表 553. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 指示重组中要完成的总工作量的值。此值可与 reorg_current_counter 配合使用以确定重组进度，reorg_current_counter 表示完成的工作量。

reorg_completion － 重组完成标志

元素标识 reorg_completion

元素类型 信息

表 554. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 表重组成功指示符。对于分区表来说，此元素还将指示数据分区的完成状态。

用法 如果表或数据分区重组操作成功，此元素的值将为 0。如果表或数据分区重组操作不成功，则此元素的值将为 -1。成功和失败值将在 sqlmon.h 中作如下定义：

- 成功：SQLM_REORG_SUCCESS
- 失败：SQLM_REORG_FAIL

如果表重组不成功，则请参阅历史记录文件以获取任何诊断信息，包括警告和错误。可使用 LIST HISTORY 命令来访问此数据。对于分区表，将对每个数据分区指示完成状态。如果索引重建在分区表上失败，则将在所有数据分区上更新失败状态。有关进一步的诊断信息，请参阅管理通知日志。

reorg_start － 表重组开始时间

元素标识 reorg_start

元素类型 时间戳记

表 555. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 表重组的开始时间。对于分区表来说，此元素还将指示每个数据分区重组的开始时间。

reorg_end － 表重组结束时间

元素标识 reorg_end

元素类型 时间戳记

表 556. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 表重组的结束时间。对于分区表来说，此元素还将指示每个数据分区重组的结束时间。

reorg_index_id － 用于重组表的索引

元素标识 reorg_index_id

元素类型 信息

表 557. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 用于重组表的索引。

reorg_tbspc_id – 用来重组表或数据分区的表空间

元素标识 reorg_tbspc_id

元素类型 信息

表 558. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 用来重组表的表空间。对于分区表来说，这将指示用来重组每个数据分区的表空间。

reorg_rows_compressed – 压缩行数

元素标识 reorg_rows_compressed

元素类型 信息

表 559. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 重组期间在表中压缩的行数。

用法 重组期间在表中压缩的行数的连续计数。某些记录永远不会被压缩（如果记录长度小于最小记录长度的话）。

重要的是要注意，这个行数未反映数据压缩效率，它只显示了符合压缩条件的记录数。

reorg_rows_rejected_for_compression – 拒绝压缩行数

元素标识 reorg_rows_rejected_for_compression

元素类型 信息

表 560. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 重组期间由于记录长度小于或等于最小记录长度而未压缩的行数。

用法 如果记录长度小于或等于最小记录长度，就不会压缩该记录。已拒绝的行数反映了这些未符合此压缩要求的记录的连续计数。

reorg_long_tbspc_id – 用来重组长对象的表空间监视元素

元素标识 reorg_long_tbspc_id

元素类型 信息

表 561. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

描述 将用来重组任何长对象（LONG VARCHAR 或 LOB 数据）的表空间。对于分区表来说，这是将用来重组每个分区的 LONG VARCHAR 和 LOB 的表空间。

SQL 游标

SQL 游标监视元素

下列元素提供有关 SQL 游标的信息:

- open_rem_curs – 打开的远程游标数监视元素
- open_rem_curs_blk – 打开的远程分块游标数监视元素
- rej_curs_blk – 拒绝的分块游标请求数监视元素
- acc_curs_blk – 接受的分块游标请求数监视元素
- open_loc_curs – 打开的本地游标数监视元素
- open_loc_curs_blk – 打开的本地游标游标数监视元素

open_rem_curs – 打开的远程游标数

元素标识 open_rem_curs

元素类型 标尺

表 562. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

描述 当前对此应用程序打开的远程游标数，包括 *open_rem_curs_blk* 计数的那些游标数。

用法 可将此元素与 *open_rem_curs_blk* 配合使用以计算作为分块游标的远程游标的百分比。如果该百分比很低，则可通过改进应用程序中的行分块来改进性能。有关更多信息，请参阅 *open_rem_curs_blk*。

有关应用程序用于连接至本地数据库的打开的游标数，请参阅 *open_loc_curs*。

相关参考:

- 第 350 页的『open_rem_curs_blk – 打开的远程分块游标数』
- 第 352 页的『open_loc_curs – 打开的本地游标数』

open_rem_curs_blk – 打开的远程分块游标数

元素标识 open_rem_curs_blk

元素类型 标尺

表 563. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

描述 当前对此应用程序打开的远程分块游标数。

用法 可将此元素与 *open_rem_curs* 配合使用以计算作为分块游标的远程游标的百分比。如果该百分比很低，则可通过改进应用程序中的行分块来改进性能：

- 检查记录分块的预编译选项以了解模糊游标的处理方式
- 重新定义游标以允许进行分块（例如，如果可能的话对游标指定 FOR FETCH ONLY）。

rej_curs_blk 和 *acc_curs_blk* 提供了附加信息，这些信息可帮助您调整配置参数以改进应用程序中的行分块。

有关应用程序用来连接至本地数据库的打开的分块游标的数目，请参阅 *open_loc_curs_blk*。

相关参考:

- 第 350 页的『open_rem_curs － 打开的远程游标数』
- 第 351 页的『rej_curs_blk － 拒绝的块游标请求数』
- 第 352 页的『acc_curs_blk － 接受的块游标请求数』
- 第 352 页的『open_loc_curs － 打开的本地游标数』
- 第 353 页的『open_loc_curs_blk － 打开的本地分块游标数』

rej_curs_blk － 拒绝的块游标请求数

元素标识 rej_curs_blk

元素类型 计数器

表 564. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 565. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

描述 在服务器上拒绝请求 I/O 块并且请求转换为非分块 I/O 的次数。

用法 如果有许多游标分块数据，则通信堆可能会变满。此堆变满时，不会返回错误。而是不会再对分块游标分配 I/O 块。如果游标无法对数据进行分块，则性能会受到影响。

如果大量游标无法执行数据分块，则可通过执行下列操作来改进性能：

- 增加 *query_heap* 数据库管理器配置参数的大小。

相关参考:

- 第 352 页的『acc_curs_blk － 接受的块游标请求数』

- 第 352 页的『open_loc_curs – 打开的本地游标数』
- 第 353 页的『open_loc_curs_blk – 打开的本地分块游标数』
- 第 350 页的『open_rem_curs – 打开的远程游标数』
- 第 350 页的『open_rem_curs_blk – 打开的远程分块游标数』

acc_curs_blk – 接受的块游标请求数

元素标识 acc_curs_blk

元素类型 计数器

表 566. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 567. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

描述 接受 I/O 块请求的次数。

用法 可将此元素与 *rej_curs_blk* 一起使用来计算接受的和／或拒绝的分块请求百分比。

有关如何使用此信息来调整配置参数的建议，请参阅 *rej_curs_blk*。

相关参考:

- 第 351 页的『rej_curs_blk – 拒绝的块游标请求数』
- 第 352 页的『open_loc_curs – 打开的本地游标数』
- 第 353 页的『open_loc_curs_blk – 打开的本地分块游标数』
- 第 350 页的『open_rem_curs – 打开的远程游标数』
- 第 350 页的『open_rem_curs_blk – 打开的远程分块游标数』

open_loc_curs – 打开的本地游标数

元素标识 open_loc_curs

元素类型 标尺

表 568. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

描述 前对此应用程序打开的本地游标数，包括 *open_loc_curs_blk* 计数的那些游标数。

用法 可将此元素与 *open_loc_curs_blk* 一起使用来计算作为分块游标的本地游标的百分比。如果该百分比很低，则可通过改进应用程序中的行分块来改进性能。

有关远程应用程序使用的游标，请参阅 *open_rem_curs*。

相关参考:

- 第 353 页的『open_loc_curs_blk – 打开的本地分块游标数』

- 第 350 页的『open_rem_curs – 打开的远程游标数』
- 第 350 页的『open_rem_curs_blk – 打开的远程分块游标数』
- 第 351 页的『rej_curs_blk – 拒绝的块游标请求数』
- 第 352 页的『acc_curs_blk – 接受的块游标请求数』

open_loc_curs_blk – 打开的本地分块游标数

元素标识 open_loc_curs_blk

元素类型 标尺

表 569. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

描述 当前对此应用程序打开的本地分块游标数。

用法 可将此元素与 *open_loc_curs* 一起使用来计算作为分块游标的本地游标的百分比。如果该百分比很低，则可通过改进应用程序中的行分块来改进性能:

- 检查记录分块的预编译选项以了解模糊游标的处理方式
- 重新定义游标以允许进行分块（例如，如果可能的话对游标指定 FOR FETCH ONLY）。

rej_curs_blk 和 acc_curs_blk 提供了附加信息，这些信息可帮助您调整配置参数以改进应用程序中的行分块。

有关远程应用程序使用的分块游标，请参阅 *open_rem_curs_blk*。

相关参考:

- 第 352 页的『open_loc_curs – 打开的本地游标数』
- 第 350 页的『open_rem_curs – 打开的远程游标数』
- 第 350 页的『open_rem_curs_blk – 打开的远程分块游标数』
- 第 351 页的『rej_curs_blk – 拒绝的块游标请求数』
- 第 352 页的『acc_curs_blk – 接受的块游标请求数』

SQL 语句活动

SQL 语句活动监视元素

下列元素提供有关 SQL 语句活动的信息:

- static_sql_stmts – 尝试的静态 SQL 语句数监视元素
- dynamic_sql_stmts – 尝试的动态 SQL 语句数监视元素
- failed_sql_stmts – 失败的语句操作数监视元素
- commit_sql_stmts – 尝试的落实语句数监视元素
- rollback_sql_stmts – 尝试的回滚语句数监视元素
- select_sql_stmts – 执行的 Select SQL 语句数监视元素
- uid_sql_stmts – 执行的 Update/Insert/Delete SQL 语句数监视元素
- ddl_sql_stmts – 数据定义语言（DDL）SQL 语句监视元素

- int_auto_rebinds － 内部自动重新绑定数监视元素
- int_commits － 内部落实数监视元素
- int_rollbacks － 内部回滚数监视元素
- int_deadlock_rollbacks － 由于死锁而导致的内部回滚数监视元素
- sql_reqs_since_commit － 自上次落实以来的 SQL 请求数监视元素
- stmt_node_number － 语句节点数监视元素
- binds_precompiles － 尝试的绑定数／预编译数监视元素

static_sql_stmts － 尝试的静态 SQL 语句数

元素标识 static_sql_stmts

元素类型 计数器

表 570. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 571. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 尝试的静态 SQL 语句数。

用法 可使用此元素来计算在数据库或应用程序级别成功执行的 SQL 语句总数:

```
  dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= 监视期间的吞吐量
```

相关参考:

- 第 355 页的『failed_sql_stmts － 失败的语句操作数』

dynamic_sql_stmts － 尝试的动态 SQL 语句数

元素标识 dynamic_sql_stmts

元素类型 计数器

表 572. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 573. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 尝试的动态 SQL 语句数。

用法 可使用此元素来计算在数据库或应用程序级别成功执行的 SQL 语句总数:

```
  dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= 监视期间的吞吐量
```

相关参考:

- 第 355 页的『failed_sql_stmts – 失败的语句操作数』

failed_sql_stmts – 失败的语句操作数

元素标识	failed_sql_stmts
元素类型	计数器

表 574. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

可将快照监视的计数器复位。

表 575. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 尝试但失败了的 SQL 语句数。

用法 可使用此元素来计算在数据库或应用程序级别成功执行的 SQL 语句总数:

```
  dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= 监视期间的吞吐量
```

此计数包括接收到负值 SQLCODE 的所有 SQL 语句数。

此元素还可帮助您确定性能低下的原因，这是因为失败的语句意味着数据库管理器浪费了时间并因而导致数据库的吞吐量很低。

相关参考:

- 第 354 页的『dynamic_sql_stmts – 尝试的动态 SQL 语句数』
- 第 354 页的『static_sql_stmts – 尝试的静态 SQL 语句数』

commit_sql_stmts – 尝试的落实语句数

元素标识 commit_sql_stmts

元素类型 计数器

表 576. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

可将快照监视的计数器复位。

表 577. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 尝试的 SQL COMMIT 语句总数。

用法 如果监视期间此计数器数字很少变化，则指示应用程序执行的落实操作较少，这可能导致登录和数据并行性出现问题。

还可使用此元素并通过计算下列各项的总和来计算工作单元总数:

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 计算的工作单元数将仅包括发生以下情况之后出现的工作单元:
- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

此计算可在数据库级别或应用程序级别完成。

相关参考:

- 第 360 页的『int_commits – 内部落实数』
- 第 357 页的『rollback_sql_stmts – 尝试的回滚语句数』
- 第 361 页的『int_rollbacks – 内部回滚数』
- 第 362 页的『int_deadlock_rollbacks – 死锁导致的内部回滚数』

rollback_sql_stmts – 尝试的回滚语句数

元素标识	rollback_sql_stmts
元素类型	计数器

表 578. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

可将快照监视的计数器复位。

表 579. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 尝试的 SQL ROLLBACK 语句总数。

用法 回滚可能是应用程序请求、死锁或错误情况导致的。此元素**仅**对从应用程序发出的回滚语句计数。

在应用程序级别，此元素可帮助您确定应用程序的数据库活动的级别以及与其他应用程序的冲突程度。在数据库级别，它可以帮助您确定数据库中的活动量以及数据库上的应用程序间的冲突程度。

注: 应尝试将回滚次数降至最低，原因是回滚活动越高，数据库的吞吐量越低。

还可使用此元素并通过计算下列各项的总和来计算工作单元总数:

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

相关参考:

- 第 365 页的『stmt_type – 语句类型』
- 第 356 页的『commit_sql_stmts – 尝试的落实语句数』
- 第 360 页的『int_commits – 内部落实数』
- 第 361 页的『int_rollbacks – 内部回滚数』
- 第 362 页的『int_deadlock_rollbacks – 死锁导致的内部回滚数』

select_sql_stmts – 执行的 SELECT SQL 语句数

元素标识	select_sql_stmts
元素类型	计数器

表 580. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
表空间	tablespace	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 581. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 执行的 SQL SELECT 语句数。

用法 可使用此元素来确定应用程序或数据库级别的数据库活动的级别。

还可使用以下公式来确定 SELECT 语句数与语句总数的比率:

```
  select_sql_stmts
/ ( static_sql_stmts
  + dynamic_sql_stmts )
```

此信息对于分析应用程序活动和吞吐量非常有用。

相关参考:

- 第 354 页的『static_sql_stmts – 尝试的静态 SQL 语句数』
- 第 354 页的『dynamic_sql_stmts – 尝试的动态 SQL 语句数』

uid_sql_stmts – 执行的 Update/Insert/Delete SQL 语句数

元素标识 uid_sql_stmts

元素类型 计数器

表 582. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 583. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 执行的 SQL UPDATE、INSERT 和 DELETE 语句数。

用法 可使用此元素来确定应用程序或数据库级别的数据库活动的级别。

还可使用以下公式来确定 UPDATE、INSERT 和 DELETE 语句数与语句总数的比率:

```
  uid_sql_stmts
 / (static_sql_stmts + dynamic_sql_stmts )
```

此信息对于分析应用程序活动和吞吐量非常有用。

相关参考:

- 第 354 页的『static_sql_stmts – 尝试的静态 SQL 语句数』
- 第 354 页的『dynamic_sql_stmts – 尝试的动态 SQL 语句数』

ddl_sql_stmts – 数据定义语言(DDL)SQL 语句数

元素标识 ddl_sql_stmts

元素类型 计数器

表 584. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 585. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 此元素指示执行的 SQL 数据定义语言(DDL)语句数。

用法 可使用此元素来确定应用程序或数据库级别的数据库活动的级别。DDL 语句的运行成本很高，这是因为它们对系统目录表有影响。因此，如果此元素的值很高，则应确定原因并在可能的情况下限制此活动的执行。

还可使用此元素并借助以下公式来确定 DDL 活动的百分比:

```
ddl_sql_stmts / 语句总数
```

此信息对于分析应用程序活动和吞吐量非常有用。DDL 语句还会影响:

- 目录高速缓存(通过使该处存储的表描述符信息和权限信息无效并导致因为从系统目录检索信息而产生其他系统开销)
- 程序包高速缓存(通过使该处存储的段无效并导致因为段重新编译而产生其他系统开销)。

DDL 语句的示例包括 CREATE TABLE、CREATE VIEW、ALTER TABLE 和 DROP INDEX。

int_auto_rebinds – 内部自动重新绑定次数

元素标识 int_auto_rebinds

元素类型 计数器

表 586. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 587. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 尝试的自动重新绑定（或重新编译）数。

用法 自动重新绑定是程序包无效时系统执行的内部绑定。当数据库管理器需要从程序包执行 SQL 语句时，将首次执行重新绑定。例如，当您执行以下操作时程序包将变得无效:

- 删除计划所依赖的对象，如表、视图或索引
- 添加或删除外键
- 撤销计划所依赖的对象特权。

可使用此元素来确定应用程序或数据库级别的数据库活动的级别。因为 int_auto_rebinds 对性能有严重影响，所以应尽可能少使用它们。

还可使用此元素并借助以下公式来确定重新绑定活动的百分比:

```
int_auto_rebinds / 语句总数
```

此信息对于分析应用程序活动和吞吐量非常有用。

相关参考:

- 第 363 页的『binds_precompiles － 尝试的绑定次数／预编译次数』

int_commits － 内部落实数

元素标识 int_commits

元素类型 计数器

表 588. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 589. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 数据库管理器在内部启动的落实总数。

用法 在执行下列任一操作期间可能发生内部落实:
- 重组
- 导入
- 绑定或预编译
- 应用程序在未执行显式 SQL COMMIT 语句的情况下结束（在 UNIX 上）。

此值（不包括显式 SQL COMMIT 语句）表示自出现下列情况后发生的内部落实数:
- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

可使用此元素并通过计算下列各项的总和来计算工作单元总数:

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 计算的工作单元数将仅包括发生以下情况之后出现的工作单元:
- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

此计算可在应用程序或数据库级别完成。

相关参考:

- 第 356 页的『commit_sql_stmts － 尝试的落实语句数』
- 第 357 页的『rollback_sql_stmts － 尝试的回滚语句数』
- 第 361 页的『int_rollbacks － 内部回滚数』

int_rollbacks － 内部回滚数

元素标识 int_rollbacks

元素类型 计数器

表 590. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 591. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 数据库管理器在内部启动的回滚总数。

用法 当下列任何一项**不能**成功完成时，将发生内部回滚:
- 重组
- 导入

- 绑定或预编译
- 应用程序因为出现死锁或锁定超时而结束
- 应用程序在未执行显式落实或回滚语句的情况下结束（在 Windows 上）。

此值表示自发生下列情况之后出现的内部回滚数:
- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

虽然此值不包括显式 SQL ROLLBACK 语句，但包括 int_deadlock_rollbacks 中的计数。

可使用此元素并通过计算下列各项的总和来计算工作单元总数:

```
   commit_sql_stmts
 + int_commits
 + rollback_sql_stmts
 + int_rollbacks
```

注: 计算的工作单元数将包括发生以下情况之后出现的工作单元:
- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

此计算可在应用程序或数据库级别完成。

相关参考:

- 第 356 页的『commit_sql_stmts – 尝试的落实语句数』
- 第 360 页的『int_commits – 内部落实数』
- 第 357 页的『rollback_sql_stmts – 尝试的回滚语句数』
- 第 362 页的『int_deadlock_rollbacks – 死锁导致的内部回滚数』

int_deadlock_rollbacks – 死锁导致的内部回滚数

元素标识 int_deadlock_rollbacks

元素类型 计数器

表 592. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 593. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

描述 数据库管理器因为死锁而启动的强制回滚总数。为解决死锁，将对数据库管理器选择的应用程序中的当前工作单元执行回滚。

用法 此元素显示已中断并且可用作并行性问题的指示符的死锁数。这很重要，原因是 int_deadlock_rollbackss 会降低数据库的吞吐量。

此值包括在 int_rollbacks 给定的值中。

相关参考:

- 第 288 页的『deadlocks – 检测到的死锁数』
- 第 357 页的『rollback_sql_stmts – 尝试的回滚语句数』
- 第 361 页的『int_rollbacks – 内部回滚数』

sql_reqs_since_commit – 上次落实后的 SQL 请求数

元素标识 sql_reqs_since_commit

元素类型 信息

表 594. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

描述 自上次落实后提交的 SQL 请求数。

用法 可使用此元素来监视事务的进度。

stmt_node_number – 语句节点

元素标识 stmt_node_number

元素类型 信息

表 595. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

描述 执行语句的节点。

用法 用于使每个语句与执行该语句的节点相关联。

binds_precompiles – 尝试的绑定次数／预编译次数

元素标识 binds_precompiles

元素类型 计数器

表 596. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 597. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 尝试的绑定次数和预编译次数。

用法 可使用此元素来了解数据库管理器内的当前活动级别。

此值不包括 *int_auto_rebinds* 的计数，但它包括因为 REBIND PACKAGE 命令而产生的绑定次数。

相关参考:

- 第 359 页的『int_auto_rebinds – 内部自动重新绑定次数』

xquery_stmts – 尝试的 XQuery 语句数

元素标识 xquery_stmts

元素类型 计数器

表 598. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 599. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

描述 已为应用程序或数据库执行的 XQuery 语句。

用法 可使用此元素来测量本地 XQuery 语言请求的活动。这不包括嵌入式 XQuery 语言请求，例如，xmlquery、xmltable 或 xmlexist。

SQL 语句详细信息

SQL 语句详细信息监视元素

注: 语句事件监视器不记录访存。

下列元素提供有关 SQL 语句的详细信息:

- stmt_type – 语句类型监视元素
- stmt_operation/operation – 语句操作监视元素
- package_name – 程序包名监视元素
- package_version_id – 程序包版本监视元素
- consistency_token – 程序包一致性标记监视元素
- section_number – 节号监视元素
- cursor_name – 游标名监视元素
- creator – 应用程序创建者监视元素
- stmt_start – 语句操作开始时间戳记监视元素
- stmt_stop – 语句操作停止时间戳记监视元素
- stop_time – 事件停止时间监视元素
- start_time – 事件开始时间监视元素
- stmt_elapsed_time – 最近的语句耗用时间监视元素

- stmt_text – SQL 动态语句文本监视元素
- stmt_sorts – 语句排序数监视元素
- fetch_count – 成功访存数监视元素
- sqlca – SQL 通信区（SQLCA）监视元素
- query_card_estimate – 估计的查询返回的行数监视元素
- query_cost_estimate – 估计的查询成本监视元素
- stmt_history_id – 语句历史记录标识监视元素stmt_history_id – 语句历史记录标识监视元素
- stmt_first_use_time – 语句的首次使用时间监视元素stmt_first_use_time – 语句的首次使用时间监视元素
- stmt_last_use_time – 语句的上次使用时间监视元素stmt_last_use_time – 语句的上次使用时间监视元素
- stmt_lock_timeout – 语句锁定超时监视元素stmt_lock_timeout – 语句锁定超时监视元素
- stmt_isolation – 语句隔离监视元素stmt_isolation – 语句隔离监视元素
- stmt_nest_level – 语句嵌套级别监视元素stmt_nest_level – 语句嵌套级别监视元素
- stmt_invocation_id – 语句调用标识监视元素stmt_invocation_id – 语句调用标识监视元素
- stmt_query_id – 语句查询标识监视元素stmt_query_id – 语句查询标识监视元素
- stmt_source_id – 语句源标识监视元素stmt_source_id – 语句源标识监视元素
- stmt_pkgcache_id – 语句程序包高速缓存标识监视元素stmt_pkgcache_id – 语句程序包高速缓存标识监视元素
- comp_env_desc – 编译环境句柄监视元素comp_env_desc – 编译环境句柄监视元素
- stmt_value_type – 值类型监视元素stmt_value_type – 值类型监视元素
- stmt_value_isnull – 值有空值监视元素stmt_value_isnull – 值有空值监视元素
- stmt_value_data – 值数据监视元素stmt_value_data – 值数据监视元素
- stmt_value_index – 值索引监视元素stmt_value_index – 值索引监视元素

stmt_type – 语句类型

元素标识 stmt_type

元素类型 信息

表 600. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 601. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-
语句	event_stmt	-

描述 所处理语句的类型。

用法 可使用此元素来确定正在执行的语句的类型。它可以是下列其中一种类型:
- 静态 SQL 语句
- 动态 SQL 语句
- SQL 语句之外的操作；如绑定或预编译操作。

对于快照监视器，此元素描述当前正在处理或最新处理的语句。

注: API 用户应参考包含数据库系统监视器常量定义的 *sqlmon.h* 头文件。

相关参考:

- 第 373 页的『stmt_text – SQL 动态语句文本』
- 第 370 页的『creator – 应用程序创建者』
- 第 369 页的『section_number – 节号』
- 第 367 页的『package_name – 程序包名称』
- 第 368 页的『package_version_id – 程序包版本』

stmt_operation/operation – 语句操作

元素标识 stmt_operation（快照监视）

operation（事件监视）

元素类型 信息

表 602. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 603. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-
语句	event_stmt	-

描述 当前处理或最新处理的语句操作（如果当前未运行任何操作）。

用法 可使用此元素来确定正在执行或最近完成的操作。

它可以是下列其中一项:

对于 SQL 操作:
- SELECT
- PREPARE
- EXECUTE
- EXECUTE IMMEDIATE
- OPEN
- FETCH
- CLOSE
- DESCRIBE
- STATIC COMMIT
- STATIC ROLLBACK

- FREE LOCATOR
- PREP_COMMIT
- CALL
- PREP_OPEN
- PREP_EXEC
- COMPILE

对于非 SQL 操作:

- RUN STATISTICS
- REORG
- REBIND
- REDISTRIBUTE
- GET TABLE AUTHORIZATION
- GET ADMINISTRATIVE AUTHORIZATION

注: API 用户应参考包含数据库系统监视器常量定义的 *sqlmon.h* 头文件。

相关参考:

- 第 365 页的『stmt_type – 语句类型』
- 第 373 页的『stmt_text – SQL 动态语句文本』
- 第 370 页的『creator – 应用程序创建者』
- 第 369 页的『section_number – 节号』
- 第 367 页的『package_name – 程序包名称』
- 第 374 页的『fetch_count – 成功的访存数』
- 第 368 页的『package_version_id – 程序包版本』

package_name – 程序包名称

元素标识 package_name

元素类型 信息

表 604. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 605. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-
语句	event_stmt	-

描述 程序包的名称，该程序包包含当前正在执行的 SQL 语句。

用法 可使用此元素来帮助标识正在执行的应用程序和 SQL 语句。

相关参考:

- 第 370 页的『creator – 应用程序创建者』
- 第 369 页的『section_number – 节号』

- 第 373 页的『stmt_text – SQL 动态语句文本』
- 第 368 页的『package_version_id – 程序包版本』

consistency_token – 程序包一致性标记

元素标识 consistency_token

元素类型 信息

表 606. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 607. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-

描述 对于给定程序包名称和创建者，可以有（从 DB2 版本 8 开始）多个版本。程序包一致性标记帮助标识包含 SQL 当前执行的程序包的版本。

用法 可使用此元素来帮助标识程序包和正在执行的 SQL 语句。

package_version_id – 程序包版本

元素标识 package_version_id

元素类型 信息

表 608. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 609. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-

描述 对于给定程序包名称和创建者，可以有（从 DB2 版本 8 开始）多个版本。程序包版本标识包含 SQL 当前执行的程序包的版本标识。程序包版本由嵌入式 SQL 程序在预编译（PREP）时使用 VERSION 关键字确定。如果在预编译时未指定，则程序包版本的值为“”（空字符串）。

用法 可使用此元素来帮助标识程序包和正在执行的 SQL 语句。

相关参考:

- 第 373 页的『stmt_text – SQL 动态语句文本』
- 第 370 页的『creator – 应用程序创建者』
- 第 367 页的『package_name – 程序包名称』
- 第 369 页的『section_number – 节号』

section_number － 节号

元素标识 section_number

元素类型 信息

表 610. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 611. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-
语句	event_stmt	-

描述 程序包中用于当前处理或最新处理的 SQL 语句的内部节号。

用法 对于静态 SQL，可将此元素与创建者、package_version_id 和 package_name 一起使用来查询 SYSCAT.STATEMENTS 系统目录表并使用以下样本查询来获取静态 SQL 语句文本：

```
SELECT SEQNO, SUBSTR(TEXT,1,120)
       FROM SYSCAT.STATEMENTS
       WHERE PKGNAME   = 'package_name' AND
             PKGSCHEMA = 'creator'      AND
             VERSION = 'package_version_id' AND
             SECTNO    = section_number
       ORDER BY SEQNO
```

注： 在获取静态语句文本时将产生警告，原因是针对系统目录表的此查询可能导致锁定争用。尽可能在没有其他针对该数据库的活动时才使用此查询。

相关参考:

- 第 373 页的『stmt_text － SQL 动态语句文本』
- 第 370 页的『creator － 应用程序创建者』
- 第 367 页的『package_name － 程序包名称』
- 第 368 页的『package_version_id － 程序包版本』

cursor_name － 游标名称

元素标识 cursor_name

元素类型 信息

表 612. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 613. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-
语句	event_stmt	-

描述 对应此 SQL 语句的游标的名称。

用法 可使用此元素来标识正在处理的 SQL 语句。将在 SQL SELECT 语句的 OPEN、FETCH、CLOSE 和 PREPARE 上使用此名称。如果未使用游标，则此字段将为空白。

相关参考:

- 第 373 页的『stmt_text – SQL 动态语句文本』
- 第 365 页的『stmt_type – 语句类型』
- 第 374 页的『fetch_count – 成功的访存数』

creator – 应用程序创建者

元素标识 creator

元素类型 信息

表 614. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 615. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
语句	event_stmt	-

描述 预编译应用程序的用户的授权标识。

用法 将此元素与目录中的程序包段信息的 CREATOR 列一起使用来帮助标识正在处理的 SQL 语句。

如果设置了 CURRENT PACKAGE PATH 专用寄存器，则 *creator* 值会在 SQL 语句有效期限内反映不同的值。如果快照或事件监视器记录是在解析 PACKAGE PATH 之前获取的，则 *creator* 值将反映客户机请求中体现的值。如果快照或事件监视器记录是在解析 PACKAGE PATH 之后获取的，则 *creator* 值将反映已解析程序包的创建者。已解析程序包将是这样一个程序包，其 *creator* 值在 CURRENT PACKAGE PATH SPECIAL REGISTER 中最早出现，并且其程序包名称和唯一标识与客户机请求的程序包名称和唯一标识相匹配。

相关参考:

- 第 367 页的『package_name – 程序包名称』
- 第 369 页的『section_number – 节号』
- 第 368 页的『package_version_id – 程序包版本』

stmt_start – 语句操作开始时间戳记

元素标识 stmt_start

元素类型 时间戳记

表 616. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

描述 stmt_operation 开始执行的日期和时间。

用法 可将此元素与 stmt_stop 一起使用来计算执行语句操作时耗用的时间。

相关参考:

- 第 371 页的『stmt_stop – 语句操作停止时间戳记』
- 第 366 页的『stmt_operation/operation – 语句操作』

stmt_stop – 语句操作停止时间戳记

元素标识 stmt_stop

元素类型 时间戳记

表 617. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

描述 stmt_operation 停止执行的日期和时间。

用法 可将此元素与 stmt_start 一起使用来计算执行语句操作时耗用的时间。

相关参考:

- 第 371 页的『stmt_start – 语句操作开始时间戳记』
- 第 366 页的『stmt_operation/operation – 语句操作』
- 第 371 页的『stop_time – 事件停止时间』

stop_time – 事件停止时间

元素标识 stop_time

元素类型 时间戳记

表 618. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	时间戳记

描述 语句停止执行的日期和时间。

用法 可将此元素与 *start_time* 配合使用来计算执行语句所耗用的时间。

对于 FETCH 语句事件，此项是上一次成功的访存操作的时间。

注: 时间戳记开关设置为 OFF 时，此元素显示为“0”。

相关参考:

- 第 371 页的『stmt_stop － 语句操作停止时间戳记』

start_time － 事件启动时间

元素标识 start_time

元素类型 时间戳记

表 619. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_start	时间戳记
事务	event_xact	时间戳记
语句	event_stmt	时间戳记
死锁	event_deadlock	时间戳记
死锁	event_dlconn	时间戳记
带有详细信息的死锁	event_detailed_dlconn	时间戳记

描述 启动工作单元、启动语句或检测死锁的日期和时间。

此元素在 event_start API 结构中指示事件监视器的启动时间。

用法 可使用此元素来将死锁连接记录与死锁事件记录相关联，并与 *stop_time* 一起使用来计算执行语句或事务所耗用的时间。

注: 时间戳记开关设置为 OFF 时，此元素显示为“0”。

相关参考:

- 第 366 页的『stmt_operation/operation － 语句操作』

stmt_elapsed_time － 最新语句耗用时间

元素标识 stmt_elapsed_time

元素类型 时间

表 620. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

描述 最新完成的语句耗用的执行时间。

用法 将此元素用作完成语句所花时间的指示符。

相关参考:

- 第 445 页的『gw_comm_errors － 通信错误数』
- 第 446 页的『gw_comm_error_time － 通信错误时间』

stmt_text – SQL 动态语句文本

元素标识 stmt_text

元素类型 信息

表 621. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
动态 SQL	dynsql	基本
DCS 语句	dcs_stmt	语句

表 622. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-
语句	event_stmt	-

描述 这是动态 SQL 语句的文本。

用法 对于应用程序快照，此语句文本帮助您标识获取快照时执行的应用程序或最新处理的应用程序（如果获取快照时正好没有在处理的语句）。

此元素返回的信息是从 SQL 语句高速缓存中获取的，并且在高速缓存溢出时可能不可用。如果要捕获语句的 SQL 文本，则唯一保险的方法是对语句使用事件监视器。

对于动态 SQL 语句，此元素标识与程序包相关联的 SQL 文本。

对于事件监视器，仅对动态语句返回此元素。如果事件监视器记录不适合事件监视器的 BUFFERSIZE，则 *stmt_text* 可能会截断以使记录能够适合。

有关如何查询系统目录表以获取因为性能注意事项而未提供的静态 SQL 语句文本的信息，请参阅 *section_number*。

相关参考:

- 第 366 页的『stmt_operation/operation – 语句操作』
- 第 369 页的『cursor_name – 游标名称』
- 第 398 页的『input_db_alias – 输入数据库别名』
- 第 370 页的『creator – 应用程序创建者』
- 第 367 页的『package_name – 程序包名称』
- 第 369 页的『section_number – 节号』
- 第 368 页的『package_version_id – 程序包版本』

stmt_sorts – 语句排序数

元素标识 stmt_sorts

元素类型 计数器

表 623. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句

表 623. 快照监视信息（续）

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
动态 SQL	dynsql	语句

描述 为处理 stmt_operation 而对一组数据排序的总次数。

用法 可使用此元素来帮助标识对索引的需求，原因是索引可以降低数据排序的需求。通过使用上表中的相关元素，您可以标识此元素为其提供排序信息的 SQL 语句，然后通过查看要排序的列（如 ORDER BY 和 GROUP BY 子句和连接列中使用的列）分析此语句以确定索引候选项。有关检查索引是否用于优化排序性能的信息，请参阅*管理指南*中的**说明**。

此计数包括数据库管理器为执行语句而在内部生成的临时表排序。排序数与 SQL 语句的第一个 FETCH 操作相关联。当语句的操作是第一个 FETCH 时，将返回此信息。您应注意到对于分块游标而言，打开游标时将执行若干次访存。在这种情况下，很难使用快照监视器来获取排序数，原因是 DB2 在内部发出第一个 FETCH 时需要获取快照。

如果要确定使用分块游标时执行的排序数，则更可靠的方法是使用对语句声明的事件监视器。CLOSE 游标的语句事件中的 total_sorts 计数器包含执行定义了游标的语句时执行的排序总数。

相关参考:

- 第 201 页的『total_sorts － 总排序数』

fetch_count － 成功的访存数

元素标识 fetch_count

元素类型 计数器

表 624. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 625. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-

描述 对于 stmt 快照监视级别和语句事件类型：对特定游标执行的成功访存数。

对于 dcs_stmt 快照监视级别：在语句执行期间尝试的物理访存数（不管应用程序访存的行数是多少）。即，fetch_count 表示处理语句时服务器需要将应答数据发送回网关的次数。

用法 可使用此元素来了解数据库管理器内的当前活动级别。

由于性能原因，语句事件监视器不会对每个 FETCH 语句生成语句事件记录。仅当 FETCH 返回非零 SQLCODE 时，才生成记录事件。

相关参考:

- 第 365 页的『stmt_type － 语句类型』
- 第 366 页的『stmt_operation/operation － 语句操作』
- 第 369 页的『cursor_name － 游标名称』
- 第 371 页的『stmt_start － 语句操作开始时间戳记』
- 第 371 页的『stmt_stop － 语句操作停止时间戳记』

sqlca － SQL 通信区（SQLCA）

元素标识 sqlca

元素类型 信息

表 626. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-

描述 在语句完成时返回至应用程序的 SQLCA 数据结构。

用法 SQLCA 数据结构可用来确定语句是否成功完成。有关 SQLCA 的内容的信息，请参阅 *SQL Reference* 或*Administrative API Reference*。

相关参考:

- 第 366 页的『stmt_operation/operation － 语句操作』

query_card_estimate － 行查询数估计

元素标识 query_card_estimate

元素类型 信息

表 627. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

描述 查询将返回的行数估计。

用法 SQL 编译器进行的此估计可与运行时实际结果进行比较。

在监视DB2 Connect时，此元素还会返回有关下列 SQL 语句的信息。

- INSERT、UPDATE 和 DELETE

  指示受影响的行数。

- PREPARE

  估计将返回的行数。仅当 DRDA 服务器为 DB2 数据库 Linux 版、UNIX 版和 Windows 版、DB2 VM 和 VSE 版或 DB2 OS/400® 版时才收集此信息。

- FETCH

  设置为访存行数。仅当 DRDA 服务器为 DB2 OS/400 版时才收集此信息。

如果不对 DRDA 服务器收集信息，则该元素设置为零。

相关参考:

- 第 376 页的『query_cost_estimate – 查询成本估计』

query_cost_estimate – 查询成本估计

元素标识 query_cost_estimate

元素类型 信息

表 628. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

描述 由 SQL 编译器确定的估计查询成本（以 timeron 计）。

用法 此项允许将实际运行时估计与编译时估计相关联。

在监视DB2 Connect时，此元素还会返回有关下列 SQL 语句的信息。

- PREPARE

  表示预编译 SQL 语句的相对成本。

- FETCH

  包含已检索行的长度。仅当 DRDA 服务器为 DB2 OS/400 版时才收集此信息。

如果不对 DRDA 服务器收集信息，则该元素设置为零。

注: 如果 DRDA 服务器为 DB2 OS/390 和 z/OS 版，则此估计可能高于 2**32 - 1（可通过不带符号的长整型变量表示的最大整数）。在此情况下，监视器对此元素返回的值将为 2**32 - 1。

stmt_history_id – 语句历史记录标识

元素标识 stmt_history_id

元素类型 信息

表 629. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史记录值	event_data_value	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

描述 此数字元素显示相对于其他语句历史记录元素而言，该语句在当前工作单元中的运行位置。在工作单元中最早运行的语句的值最低。如果同一语句在同一工作单元中运行两次，则该语句在两个不同位置出现时将显示两个不同的 stmt_history_id 值。

用法 可使用此信息来了解导致死锁的 SQL 语句的顺序。

stmt_first_use_time – 语句第一次使用时间

元素标识 stmt_first_use_time

元素类型 信息

表 630. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	时间戳记
带有详细信息的死锁的历史纪录	event_stmt_history	时间戳记

描述 此元素显示第一次处理语句条目的时间。对于游标操作，stmt_first_use_time 将显示打开游标的时间。在应用程序协调节点，此值反映应用程序请求；在非协调程序节点，此值反映从原始节点接收请求的时间。

用法 将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_last_use_time – 语句上一次使用时间监视元素

元素标识 stmt_last_use_time

元素类型 信息

表 631. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	时间戳记
带有详细信息的死锁的历史纪录	event_stmt_history	时间戳记

描述 此元素显示上一次处理语句条目的时间。对于游标操作，stmt_last_use_time 显示操作可能为打开、访存或关闭的游标的上一次操作的时间。在应用程序协调节点，此值反映应用程序请求；在非协调程序节点，此值反映从原始节点接收请求的时间。

用法 将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_lock_timeout – 语句锁定超时

元素标识 stmt_lock_timeout

元素类型 信息

表 632. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

描述 此元素显示运行语句时对该语句生效的锁定超时值。

用法 可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因和特定 SQL 语句的执行行为。

stmt_isolation – 语句隔离

元素标识 stmt_isolation

元素类型 信息

表 633. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

描述 此元素显示运行语句时对该语句生效的隔离值。

可能的隔离级别值列表如下所示:

- SQLM_ISOLATION_LEVEL_NONE 0（未指定隔离级别）
- SQLM_ISOLATION_LEVEL_UR 1（未落实的读）
- SQLM_ISOLATION_LEVEL_CS 2（游标稳定性）
- SQLM_ISOLATION_LEVEL_RS 3（读稳定性）
- SQLM_ISOLATION_LEVEL_RR 4（可重复读）

用法 可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因和特定 SQL 语句的执行行为。

stmt_nest_level – 语句嵌套级别

元素标识 stmt_nest_level

元素类型 信息

表 634. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

描述 此元素显示运行语句时生效的嵌套或递归的级别; 每个嵌套级别对应一个存储过程或用户定义的函数（UDF）的嵌套或递归调用。

用法 可将此元素与 stmt_invocation_id 一起使用来唯一标识特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_invocation_id – 语句调用标识

元素标识 stmt_invocation_id

元素类型 信息

表 635. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

描述 此元素显示运行 SQL 语句的例程调用的标识。该值指示在当前嵌套级别进行的例程调用次数，这些调用是该级别在应用程序中活动时进行的。

用法 可将此元素与 stmt_nest_level 一起使用来唯一标识特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_query_id － 语句查询标识

元素标识 stmt_query_id

元素类型 信息

表 636. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

描述 此元素显示对用作游标的任何 SQL 语句指定的内部查询标识。

用法 可将此元素与 stmt_nest_level 一起使用来唯一标识特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_source_id － 语句源标识

元素标识 stmt_source_id

元素类型 信息

表 637. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

描述 此元素显示对运行的 SQL 语句的源指定的内部标识。

用法 可将此元素与 appl_id 一起使用来唯一标识运行特定 SQL 语句的请求的源。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_pkgcache_id – 语句程序包高速缓存标识

元素标识 stmt_pkgcache_id

元素类型 信息

表 638. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

描述 此元素显示动态 SQL 语句的内部程序包高速缓存标识。

用法 可使用此元素来唯一标识特定 SQL 语句。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

comp_env_desc – 编译环境句柄

元素标识 comp_env_desc

元素类型 信息

表 639. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

描述 此元素表示编译 SQL 语句时使用的编译环境的句柄。

用法 可提供此元素作为 COMPILATION_ENV 表函数的输入，或者作为 SET COMPILATION ENVIRONMENT SQL 语句的输入。

stmt_value_type – 值类型

元素标识 stmt_value_type

元素类型 信息

表 640. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	stmt_value_type	-

描述 此元素包含与 SQL 语句相关联的数据值类型的字符串表示。

用法 可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_value_isnull – 包含空值

元素标识 stmt_value_isnull

元素类型 信息

表 641. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	stmt_value_isnull	-

描述 此元素显示与 SQL 语句相关联的数据值是否为空值。

用法 可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_value_data – 值数据

元素标识 stmt_value_data

元素类型 信息

表 642. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	stmt_value_data	-

描述 此元素包含 SQL 语句的数据值的字符串表示。LOB、LONG、DATALINK 和结构化类型参数显示为空字符串。日期、时间和时间戳记字段记录为 ISO 格式。

用法 可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_value_index – 值索引

元素标识 stmt_value_index

元素类型 信息

表 643. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	stmt_value_data	-

描述 此元素表示 SQL 语句中使用的输入参数标记或主变量的位置。

用法 可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

inact_stmthist_sz – 语句历史记录列表大小

元素标识 inact_stmthist_sz

元素类型 信息

表 644. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	-
database	db	-

描述 当带有历史记录的详细信息死锁事件监视器运行时，此元素将报告数据库监视器堆（MON_HEAP_SZ）中用于记录语句历史记录列表项的字节数。

用法 可在调整数据库监视器堆时使用此元素。

stmt_value_isreoptvalue – 用于语句重新优化的变量

元素标识 stmt_value_isreoptvalue

元素类型 信息

表 645. 快照监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	stmt_value_isreoptvalue	-
database	db	-

描述 此元素显示是否在语句重新优化期间使用了提供的值。如果语句重新优化（例如，因为设置 REOPT 绑定选项），或者该值在此重新优化期间用作 SQL 编译器的输入，则返回值 『True』。

用法 可以将此元素与提供的编译环境一起使用，以允许就 SQL 编译器对 SQL 语句的处理进行完整的分析。

子节详细信息

子节详细信息监视元素

对分区数据库执行语句时，它将分为若干子节，并且这些子节可能在不同分区上执行。应用程序可能会让若干子节同时在一个分区上运行。

对于问题确定，可能必须找到问题子节。例如，某个子节可能正在等待表队列，原因是要写至此表队列的其中一个写程序在另一节点上处于锁定等待状态。要了解应用程序的整体情况，可能必须对运行应用程序的每个节点发出应用程序快照。

下列数据库系统监视器元素提供有关子节的信息:
- ss_number – 子节号监视元素
- ss_node_number – 子节节点号监视元素
- ss_status – 子节状态监视元素
- ss_exec_time – 执行子节耗用时间监视元素
- tq_wait_for_any – 等待任何节点在表队列上发送监视元素
- tq_node_waited_for – 等待表队列上的节点监视元素
- tq_tot_send_spills – 溢出的表队列缓冲区总数监视元素
- tq_cur_send_spills – 溢出的表队列缓冲区当前数目监视元素
- tq_rows_read – 从表队列读取的行数监视元素
- tq_rows_written – 写入表队列的行数监视元素
- tq_max_send_spills – 最大表队列缓冲区溢出数监视元素
- tq_id_waiting_on – 等待表队列上的节点监视元素

ss_number – 子节号

元素标识 ss_number

元素类型 信息

表 646. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 647. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

描述 标识与返回的信息相关联的子节。

用法 此编号与访问方案中可使用 db2expln 获取的子节号相关联。

ss_node_number – 子节节点号

元素标识 ss_node_number

元素类型 信息

表 648. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 649. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

描述 执行子节的节点。

用法 用于使每个子节与执行该子节的数据库分区相关联。

ss_status – 子节状态

元素标识 ss_status

元素类型 信息

表 650. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

描述 正在执行的子节的当前状态。

用法 当前状态值可能是:
- 正在执行（sqlmon.h 中的 SQLM_SSEXEC）
- 等待锁定
- 在表队列上等待接收数据
- 在表队列上等待发送数据

相关参考:

- 第 384 页的『tq_node_waited_for – 在表队列上等待节点』
- 第 384 页的『tq_wait_for_any – 在表队列上等待发送任何节点』

ss_exec_time － 子节执行耗用时间

元素标识 ss_exec_time

元素类型 计数器

表 651. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 652. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

描述 执行子节所花的时间（以秒计）。

用法 允许您跟踪子节进度。

tq_wait_for_any － 在表队列上等待发送任何节点

元素标识 tq_wait_for_any

元素类型 信息

表 653. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

描述 此标志用于指示子节已阻塞，原因是它正在等待从任何节点接收行。

用法 如果 ss_status 指示*在表队列上等待接收数据*并且此标志为 TRUE，则表示子节正等待从任何节点接收行。这通常指示 SQL 语句未处理至可将数据传递至等待代理程序的点。例如，写代理程序可能正在执行排序并且在排序完成之前不会写入行。从 db2expln 输出来确定与表队列相关联的子节号，代理程序正在等待接收来自该表队列的行。然后可通过在执行子节的每个节点上获取快照来检查该子节的状态。

相关参考:

- 第 383 页的『ss_status － 子节状态』
- 第 384 页的『tq_node_waited_for － 在表队列上等待节点』

tq_node_waited_for － 在表队列上等待节点

元素标识 tq_node_waited_for

元素类型 信息

表 654. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

描述 如果子节状态 ss_status 为*等待接收*或*等待发送*并且 tq_wait_for_any 为 FALSE，则此项是此代理程序正在等待的节点的编号。

用法 它可以用于故障诊断。您可能想要在子节正在等待的节点上获取应用程序快照。例如，应用程序在该节点上可能处于锁定等待状态。

相关参考:

- 第 383 页的『ss_status – 子节状态』
- 第 384 页的『tq_wait_for_any – 在表队列上等待发送任何节点』

tq_tot_send_spills – 溢出的表队列缓冲区总数

元素标识 tq_tot_send_spills

元素类型 计数器

表 655. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 656. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

描述 溢出至临时表的表队列缓冲区总数。

用法 指示已写至临时表的表队列缓冲区总数。有关更多信息，请参阅 tq_cur_send_spills。

相关参考:

- 第 383 页的『ss_status – 子节状态』
- 第 385 页的『tq_cur_send_spills – 当前溢出的表队列缓冲区数』

tq_cur_send_spills – 当前溢出的表队列缓冲区数

元素标识 tq_cur_send_spills

元素类型 标尺

表 657. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

描述 当前驻留在临时表中的表队列缓冲区数。

用法 写至表队列的代理程序可能正将行发送至若干阅读器。当向其发送行的代理程序不接受行而另一代理程序需要行以进行处理时，写代理程序会将缓冲区溢出至临时表。溢出至临时表允许写程序和其他阅读器同时继续进行处理。

当读取代理程序准备接受更多行时，已溢出的行将发送至该代理程序。

如果此数字很高，并且查询失败（其 sqlcode 为 -968），同时 *db2diad.log* 中的消息指示 TEMP 表空间中的临时空间已用完，则可能是表队列溢出造成的。这可能指示另一节点存在问题（如锁定）。应通过在所有分区上获取此查询的快照来进行调查。

还可能出现下列情况：因为数据的分区方式而使得许多缓冲区需要溢出以供该查询使用。在这类情况下，您需要为临时表空间添加更多磁盘。

相关参考:

- 第 383 页的『ss_status – 子节状态』
- 第 385 页的『tq_tot_send_spills – 溢出的表队列缓冲区总数』

tq_rows_read – 从表队列读取的行数

元素标识 tq_rows_read

元素类型 计数器

表 658. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 659. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

描述 从表队列读取的总行数。

用法 如果监视未指示此数字正在增加，则表示处理未在进行。

如果此数字在节点间有很大差别，则表示一些节点可能使用过度而另一些节点使用得不多。

如果此数字很大，则表示节点间传递的数据很多，建议进行优化以改进访问方案。

tq_rows_written – 写至表队列的行数

元素标识 tq_rows_written

元素类型 计数器

表 660. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 661. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

描述 写至表队列的总行数。

用法 如果监视未指示此数字正在增加，则表示处理未在进行。

如果此数字在节点间有很大差别，则表示一些节点可能使用过度而另一些节点使用得不多。

如果此数字很大，则表示节点间传递的数据很多，建议进行优化以改进访问方案。

tq_max_send_spills － 最大表队列缓冲区溢出数

元素标识	tq_max_send_spills
元素类型	水位标记

表 662. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 663. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

描述 溢出至临时表的最大表队列缓冲区数。

用法 指示已写至临时表的最大表队列缓冲区数。

相关参考:

- 第 385 页的『tq_tot_send_spills － 溢出的表队列缓冲区总数』
- 第 385 页的『tq_cur_send_spills － 当前溢出的表队列缓冲区数』

tq_id_waiting_on － 在表队列的节点上等待

元素标识	tq_id_waiting_on
元素类型	信息

表 664. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

描述 正在等待的代理程序。

用法 它可以用于故障诊断。

相关参考:

- 第 383 页的『ss_status － 子节状态』
- 第 384 页的『tq_node_waited_for － 在表队列上等待节点』

动态 SQL

动态 SQL 监视元素

DB2 语句高速缓存为经常使用的 SQL 语句存储程序包和统计信息。通过检查此高速缓存的内容，可标识使用得最频繁的动态 SQL 语句及消耗最多资源的查询。使用此信息，可检查执行得最多的和执行成本最高的 SQL 操作，以确定进行 SQL 调整是否能获取更好的数据库性能。

- num_executions － 语句执行数监视元素
- num_compilations － 语句编译数监视元素
- prep_time_worst － 最长的语句准备时间监视元素

- prep_time_best － 最短的语句准备时间监视元素
- total_exec_time － 耗用的语句执行时间监视元素

num_executions － 语句执行次数

元素标识 num_executions

元素类型 计数器

表 665. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

可将快照监视的计数器复位。

描述 已执行 SQL 语句的次数。

用法 可使用此元素来标识系统中执行得最频繁的 SQL 语句。

相关参考:

- 第 388 页的『num_compilations － 语句编译次数』

num_compilations － 语句编译次数

元素标识 num_compilations

元素类型 计数器

表 666. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

描述 特定 SQL 语句的不同编译的次数。

用法 对于某些对不同模式发出的 SQL 语句（例如“select t1 from foo”）来说，尽管它们引用不同的访问方案，但却表现为在 DB2 高速缓存中的同一个语句中。将此值与 num_executions 配合使用，以确定不佳编译环境是否会导致动态 SQL 快照统计信息的结果出现偏差。

相关参考:

- 第 388 页的『num_executions － 语句执行次数』

prep_time_worst － 语句最长编译时间

元素标识 prep_time_worst

元素类型 水位标记

表 667. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

描述 编译特定 SQL 语句所需的最长时间（以微秒计）。

用法 将此值与 prep_time_best 配合使用来标识编译成本高昂的 SQL 语句。

相关参考:

- 第 389 页的『prep_time_best － 语句最短编译时间』

prep_time_best － 语句最短编译时间

元素标识 prep_time_best

元素类型 水位标记

表 668. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

描述 编译特定 SQL 语句所需的最短时间。

用法 将此值与 prep_time_worst 配合使用来标识编译成本高昂的 SQL 语句。

相关参考:

- 第 388 页的『prep_time_worst － 语句最长编译时间』

total_exec_time － 执行语句所耗用的时间

元素标识 total_exec_time

元素类型 时间

表 669. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

描述 在 SQL 高速缓存中执行特定语句所花的总时间（以秒和微秒计）。

用法 将此元素与 num_executions 配合使用来确定语句的平均耗用时间并标识调整 SQL 时受益最多的 SQL 语句。在评估此元素的内容时，必须考虑 num_compilation。

相关参考:

- 第 388 页的『num_executions － 语句执行次数』
- 第 388 页的『num_compilations － 语句编译次数』
- 第 396 页的『total_sys_cpu_time － 语句的总系统 CPU 』
- 第 397 页的『total_usr_cpu_time － 语句的总用户 CPU 』

查询内并行性

查询内并行性监视元素

下列数据库系统监视器元素提供有关并行度大于 1 的查询的信息:

- num_agents － 处理语句的代理程序数监视元素
- agents_top － 创建的代理程序数监视元素
- degree_parallelism － 并行度监视元素

num_agents － 正在处理语句的代理程序数

元素标识 num_agents

元素类型 标尺

表 670. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
应用程序	subsection	语句

描述 当前执行语句或子节的并行代理程序数。

用法 指示查询并行度的指示符。此项对于通过获取连续快照来跟踪查询执行进度非常有用。

相关参考:

- 第 390 页的『agents_top － 创建的代理程序数』
- 第 390 页的『degree_parallelism － 并行度』

agents_top － 创建的代理程序数

元素标识 agents_top

元素类型 水位标记

表 671. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句
应用程序	stmt	语句

描述 在应用程序级别，此项是执行语句时使用的代理程序的最大数目。在数据库级别，此项是用于所有应用程序的代理程序的最大数目。

用法 指示查询内并行性实现情况的指示符。

相关参考:

- 第 390 页的『num_agents － 正在处理语句的代理程序数』
- 第 390 页的『degree_parallelism － 并行度』

degree_parallelism － 并行度

元素标识 degree_parallelism

元素类型 信息

表 672. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

描述 绑定查询时请求的并行度。

用法 与 agents_top 配合使用来确定查询是否达到最大级别的并行性。

相关参考:

- 第 390 页的『num_agents – 正在处理语句的代理程序数』
- 第 390 页的『agents_top – 创建的代理程序数』

CPU 使用情况

CPU 使用情况监视元素

应用程序的 CPU 使用情况分为**用户 CPU** 和**系统 CPU**，用户 CPU 是执行应用程序代码时消耗的 CPU，而系统 CPU 是执行系统调用时消耗的 CPU。

可在应用程序、事务、语句和子节级别提供 CPU 消耗情况。
- agent_usr_cpu_time – 代理程序使用的用户 CPU 时间监视元素
- agent_sys_cpu_time – 代理程序使用的系统 CPU 时间监视元素
- stmt_usr_cpu_time – 语句使用的用户 CPU 时间监视元素
- stmt_sys_cpu_time – 语句使用的系统 CPU 时间监视元素
- user_cpu_time – 用户 CPU 时间监视元素
- system_cpu_time – 系统 CPU 时间监视元素
- ss_usr_cpu_time – 子节所使用的用户 CPU 时间监视元素
- ss_sys_cpu_time – 子节所使用的系统 CPU 时间监视元素
- total_sys_cpu_time – 用于语句的总系统 CPU 监视元素
- total_usr_cpu_time – 用于语句的总用户 CPU 监视元素

agent_usr_cpu_time – 代理程序使用的用户 CPU 时间

元素标识 agent_usr_cpu_time

元素类型 时间

表 673. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

可将快照监视的计数器复位。

描述 数据库管理器代理进程使用的总 CPU 时间（以秒和微秒计）。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您标识消耗大量 CPU 的应用程序或查询。

此计数器包括花费在 SQL 和非 SQL 语句上的时间，同时包括花费在应用程序执行的所有不受防护的用户定义的函数（UDF）或存储过程上的时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，则此元素将返回为 0。

相关参考:

- 第 392 页的『agent_sys_cpu_time – 代理程序使用的系统 CPU 时间』
- 第 392 页的『stmt_usr_cpu_time – 语句使用的用户 CPU 时间』
- 第 393 页的『stmt_sys_cpu_time – 语句使用的系统 CPU 时间』

- 第 395 页的『ss_usr_cpu_time – 子节使用的用户 CPU 时间』
- 第 396 页的『ss_sys_cpu_time – 子节使用的系统 CPU 时间』
- 第 394 页的『user_cpu_time – 用户 CPU 时间』
- 第 394 页的『system_cpu_time – 系统 CPU 时间』
- 第 396 页的『total_sys_cpu_time – 语句的总系统 CPU 』
- 第 397 页的『total_usr_cpu_time – 语句的总用户 CPU 』

agent_sys_cpu_time – 代理程序使用的系统 CPU 时间

元素标识 agent_sys_cpu_time

元素类型 时间

表 674. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

对于应用程序级别的快照监视，可复位此计数器。不能在其他级别复位此计数器。

描述 数据库管理器代理进程使用的总系统 CPU 时间（以秒和微秒计）。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

此元素包括花费在 SQL 和非 SQL 语句上的 CPU 时间，同时包括花费在所有不受防护的用户定义的函数（UDF）上的 CPU 时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，则此元素将设置为 0。

相关参考:

- 第 391 页的『agent_usr_cpu_time – 代理程序使用的用户 CPU 时间』
- 第 392 页的『stmt_usr_cpu_time – 语句使用的用户 CPU 时间』
- 第 393 页的『stmt_sys_cpu_time – 语句使用的系统 CPU 时间』
- 第 395 页的『ss_usr_cpu_time – 子节使用的用户 CPU 时间』
- 第 396 页的『ss_sys_cpu_time – 子节使用的系统 CPU 时间』
- 第 394 页的『user_cpu_time – 用户 CPU 时间』
- 第 394 页的『system_cpu_time – 系统 CPU 时间』
- 第 396 页的『total_sys_cpu_time – 语句的总系统 CPU 』
- 第 397 页的『total_usr_cpu_time – 语句的总用户 CPU 』

stmt_usr_cpu_time – 语句使用的用户 CPU 时间

元素标识 stmt_usr_cpu_time

元素类型 时间

表 675. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句，时间戳记
应用程序	stmt	语句，时间戳记

描述 当前执行的语句使用的总*用户* CPU 时间（以秒和微秒计）。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

此计数器包括花费在 SQL 和非 SQL 语句上的时间，同时包括花费在应用程序执行的所有不受防护的用户定义的函数（UDF）或存储过程上的时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，则此元素将设置为 0。

相关参考:

- 第 392 页的『agent_sys_cpu_time – 代理程序使用的系统 CPU 时间』
- 第 391 页的『agent_usr_cpu_time – 代理程序使用的用户 CPU 时间』
- 第 393 页的『stmt_sys_cpu_time – 语句使用的系统 CPU 时间』
- 第 395 页的『ss_usr_cpu_time – 子节使用的用户 CPU 时间』
- 第 396 页的『ss_sys_cpu_time – 子节使用的系统 CPU 时间』
- 第 394 页的『user_cpu_time – 用户 CPU 时间』
- 第 394 页的『system_cpu_time – 系统 CPU 时间』
- 第 396 页的『total_sys_cpu_time – 语句的总系统 CPU 』
- 第 397 页的『total_usr_cpu_time – 语句的总用户 CPU 』

stmt_sys_cpu_time – 语句使用的系统 CPU 时间

元素标识 stmt_sys_cpu_time

元素类型 时间

表 676. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句，时间戳记
应用程序	stmt	语句，时间戳记

描述 当前执行的语句使用的总*系统* CPU 时间（以秒和微秒计）。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

此计数器包括花费在 SQL 和非 SQL 语句上的时间，同时包括花费在应用程序执行的所有不受防护的用户定义的函数（UDF）或存储过程上的时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，则此元素将设置为 0。

相关参考:

- 第 392 页的『agent_sys_cpu_time – 代理程序使用的系统 CPU 时间』
- 第 392 页的『stmt_usr_cpu_time – 语句使用的用户 CPU 时间』
- 第 391 页的『agent_usr_cpu_time – 代理程序使用的用户 CPU 时间』
- 第 395 页的『ss_usr_cpu_time – 子节使用的用户 CPU 时间』
- 第 396 页的『ss_sys_cpu_time – 子节使用的系统 CPU 时间』
- 第 394 页的『user_cpu_time – 用户 CPU 时间』
- 第 394 页的『system_cpu_time – 系统 CPU 时间』
- 第 396 页的『total_sys_cpu_time – 语句的总系统 CPU 』
- 第 397 页的『total_usr_cpu_time – 语句的总用户 CPU 』

user_cpu_time – 用户 CPU 时间

元素标识 user_cpu_time

元素类型 时间

表 677. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
事务	event_xact	-
语句	event_stmt	-

描述 数据库管理器代理进程、工作单元或语句使用的总*用户* CPU 时间（以秒和微秒计）。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

注: 如果此信息对您的操作系统不可用，则此元素将设置为 0。

相关参考:

- 第 392 页的『agent_sys_cpu_time – 代理程序使用的系统 CPU 时间』
- 第 392 页的『stmt_usr_cpu_time – 语句使用的用户 CPU 时间』
- 第 393 页的『stmt_sys_cpu_time – 语句使用的系统 CPU 时间』
- 第 395 页的『ss_usr_cpu_time – 子节使用的用户 CPU 时间』
- 第 396 页的『ss_sys_cpu_time – 子节使用的系统 CPU 时间』
- 第 391 页的『agent_usr_cpu_time – 代理程序使用的用户 CPU 时间』
- 第 394 页的『system_cpu_time – 系统 CPU 时间』
- 第 396 页的『total_sys_cpu_time – 语句的总系统 CPU 』
- 第 397 页的『total_usr_cpu_time – 语句的总用户 CPU 』

system_cpu_time – 系统 CPU 时间

元素标识 system_cpu_time

元素类型 时间

表 678. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
事务	event_xact	-
语句	event_stmt	-

描述 数据库管理器代理进程、工作单元或语句使用的总系统 CPU 时间（以秒和微秒计）。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为其他调整而受益的应用程序。

注： 如果此信息对您的操作系统不可用，则此元素将设置为 0。

相关参考:

- 第 392 页的『agent_sys_cpu_time – 代理程序使用的系统 CPU 时间』
- 第 392 页的『stmt_usr_cpu_time – 语句使用的用户 CPU 时间』
- 第 393 页的『stmt_sys_cpu_time – 语句使用的系统 CPU 时间』
- 第 395 页的『ss_usr_cpu_time – 子节使用的用户 CPU 时间』
- 第 396 页的『ss_sys_cpu_time – 子节使用的系统 CPU 时间』
- 第 394 页的『user_cpu_time – 用户 CPU 时间』
- 第 391 页的『agent_usr_cpu_time – 代理程序使用的用户 CPU 时间』
- 第 396 页的『total_sys_cpu_time – 语句的总系统 CPU 』
- 第 397 页的『total_usr_cpu_time – 语句的总用户 CPU 』

ss_usr_cpu_time – 子节使用的用户 CPU 时间

元素标识	ss_usr_cpu_time
元素类型	时间

表 679. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	时间戳记

表 680. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	时间戳记

描述 当前执行的语句子节使用的总用户 CPU 时间（以秒和微秒计）。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

相关参考:

- 第 392 页的『agent_sys_cpu_time – 代理程序使用的系统 CPU 时间』

- 第 392 页的『stmt_usr_cpu_time – 语句使用的用户 CPU 时间』
- 第 393 页的『stmt_sys_cpu_time – 语句使用的系统 CPU 时间』
- 第 391 页的『agent_usr_cpu_time – 代理程序使用的用户 CPU 时间』
- 第 396 页的『ss_sys_cpu_time – 子节使用的系统 CPU 时间』
- 第 394 页的『user_cpu_time – 用户 CPU 时间』
- 第 394 页的『system_cpu_time – 系统 CPU 时间』
- 第 396 页的『total_sys_cpu_time – 语句的总系统 CPU 』
- 第 397 页的『total_usr_cpu_time – 语句的总用户 CPU 』

ss_sys_cpu_time – 子节使用的系统 CPU 时间

元素标识 ss_sys_cpu_time

元素类型 时间

表 681. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	时间戳记

表 682. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	时间戳记

描述 当前执行的语句子节使用的总系统 CPU 时间（以秒和微秒计）。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

相关参考:

- 第 392 页的『agent_sys_cpu_time – 代理程序使用的系统 CPU 时间』
- 第 392 页的『stmt_usr_cpu_time – 语句使用的用户 CPU 时间』
- 第 393 页的『stmt_sys_cpu_time – 语句使用的系统 CPU 时间』
- 第 395 页的『ss_usr_cpu_time – 子节使用的用户 CPU 时间』
- 第 394 页的『user_cpu_time – 用户 CPU 时间』
- 第 391 页的『agent_usr_cpu_time – 代理程序使用的用户 CPU 时间』
- 第 396 页的『total_sys_cpu_time – 语句的总系统 CPU 』
- 第 397 页的『total_usr_cpu_time – 语句的总用户 CPU 』

total_sys_cpu_time – 语句的总系统 CPU

元素标识 total_sys_cpu_time

元素类型 时间

表 683. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

描述 SQL 语句的总系统 CPU 时间。

用法 将此元素与耗用的语句执行时间和语句的总用户 CPU 配合使用以找出执行成本最高的语句。

相关参考:

- 第 397 页的『total_usr_cpu_time － 语句的总用户 CPU 』
- 第 389 页的『total_exec_time － 执行语句所耗用的时间』

total_usr_cpu_time － 语句的总用户 CPU

元素标识 total_usr_cpu_time

元素类型 时间

表 684. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

描述 SQL 语句的总用户 CPU 时间。

用法 将此元素与耗用的语句执行时间配合使用以找出运行时间最长的语句。

相关参考:

- 第 396 页的『total_sys_cpu_time － 语句的总系统 CPU 』
- 第 389 页的『total_exec_time － 执行语句所耗用的时间』

快照监视

快照监视监视元素

下列元素提供有关监视应用程序的信息。它们是作为每个快照的输出返回的:
- last_reset － 最近的复位时间戳记监视元素
- input_db_alias － 输入数据库别名监视元素
- time_stamp － 快照时间监视元素
- num_nodes_in_db2_instance － 分区中的节点数监视元素

last_reset － 最后复位时间戳记

元素标识 last_reset

元素类型 时间戳记

表 685. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	时间戳记

表 685. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
数据库	dbase	时间戳记
应用程序	appl	时间戳记
表空间	tablespace_list	缓冲池，时间戳记
表	table_list	时间戳记
DCS 数据库	dcs_dbase	时间戳记
DCS 应用程序	dcs_appl	时间戳记

描述 指示监视器计数器对发出 GET SNAPSHOT 的应用程序复位的日期和时间。

用法 可使用此元素来帮助定义数据库系统监视器返回的信息的作用域。

如果计数器从未复位，则此元素将为零。

仅当复位所有活动数据库时，数据库管理器计数器才会复位。

相关参考:

- 第 398 页的『input_db_alias – 输入数据库别名』

input_db_alias – 输入数据库别名

元素标识 input_db_alias

元素类型 信息

表 686. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl_id_info	基本
表空间	tablespace_list	缓冲池
缓冲池	bufferpool	缓冲池
表	table_list	表
锁	db_lock_list	基本

描述 调用快照函数时提供的数据库别名。

用法 此元素可用来标识监视器数据适用的特定数据库。除非请求与特定数据库相关的监视器信息，否则它将包含空白。

此字段的值可能不同于 *client_db_alias* 监视元素的值，原因是数据库可能具有许多不同别名。不同应用程序和用户可使用不同别名来连接至同一数据库。

相关参考:

- 第 397 页的『last_reset – 最后复位时间戳记』
- 第 168 页的『client_db_alias – 应用程序使用的数据库别名』

time_stamp – 快照时间

元素标识 time_stamp

元素类型 时间戳记

表 687. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

描述 收集数据库系统监视器信息的日期和时间。

用法 如果要将结果保存在文件或数据库中以便将来进行分析，则可使用此元素来帮助将相关数据按时间顺序排列。

num_nodes_in_db2_instance － 分区中的节点数

元素标识 num_nodes_in_db2_instance

元素类型 信息

表 688. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

表 689. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

描述 获取快照的实例上的节点数。

用法 使用此元素来确定实例的节点数。对于非分区系统数据库，此值将为 1。

事件监视

事件监视监视元素

下列元素提供有关监视应用程序的信息。它们是作为事件的输出返回的:

- count － 事件监视器溢出数监视元素
- first_overflow_time － 首次事件溢出的时间监视元素
- last_overflow_time － 最后一次事件溢出的时间监视元素
- byte_order － 事件数据的字节顺序监视元素
- version － 监视器数据的版本监视元素
- event_monitor_name － 事件监视器名监视元素
- partial_record － 部分记录监视元素
- event_time － 事件时间监视元素
- evmon_flushes － 事件监视器的清空次数监视元素
- evmon_activates － 事件监视器的激活次数监视元素
- sql_req_id － SQL 语句的请求标识监视元素
- message － 控制表消息监视元素
- message_time － 时间戳记控制表消息监视元素
- partition_number － 分区号监视元素

count － 事件监视器溢出数

元素标识 计数

元素类型 计数器

表 690. 事件监视信息

事件类型	逻辑数据分组	监视开关
溢出记录	event_overflow	-

描述 发生的连续溢出数。

用法 可使用此元素来了解丢失的监视器数据量。

事件监视器对一组连续溢出发送一个溢出记录。

first_overflow_time － 第一次事件溢出时间

元素标识 first_overflow_time

元素类型 时间戳记

表 691. 事件监视信息

事件类型	逻辑数据分组	监视开关
溢出记录	event_overflow	-

描述 此溢出记录记录的第一次溢出的日期和时间。

用法 将此元素与 *last_over_flow time* 配合使用来计算生成溢出记录所耗用的时间。

相关参考:

- 第 400 页的『count － 事件监视器溢出数』

last_overflow_time － 最后一次事件溢出时间

元素标识 last_overflow_time

元素类型 时间戳记

表 692. 事件监视信息

事件类型	逻辑数据分组	监视开关
溢出记录	event_overflow	-

描述 此溢出记录记录的最后一次溢出的日期和时间。

用法 将此元素与 *first_overflow_time* 配合使用来计算生成溢出记录所耗用的时间。

相关参考:

- 第 400 页的『count － 事件监视器溢出数』

byte_order － 事件数据的字节顺序

元素标识 byte_order

元素类型 信息

表 693. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

描述 数字数据的字节定序，具体地说是在“大尾数法”服务器（如 RISC System/6000®）还是“小尾数法”服务器（如基于 Intel® 并且运行 Windows 2000 的 PC）上生成事件数据流。

用法 因为“大尾数法”服务器上的整数字节顺序与“小尾数法”服务器上的字节顺序方向相反，所以必须使用此信息以允许您解释数据流中的数字数据。

如果处理数据的应用程序识别它在一种类型的计算机硬件（如大尾数法计算机）上运行，而事件数据是在另一种类型的计算机硬件（如小尾数法计算机）上生成的，则监视应用程序必须先使数字数据字段的字节反向，然后再解释它们。否则不需要进行字节定向。

此元素可设置为下列其中一种 API 常量:
- SQLM_BIG_ENDIAN
- SQLM_LITTLE_ENDIAN

version – 监视器数据版本

元素标识 版本

元素类型 信息

表 694. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

描述 产生事件监视器数据流的数据库管理器版本。

用法 事件监视器使用的数据结构在数据库管理器发行版之间可能更改。因此，监视器应用程序应检查数据流版本以确定它们能否处理要接收的数据。

对于此发行版，此元素设置为 API 常量 SQLM_DBMON_VERSION8。

event_monitor_name – 事件监视器名称

元素标识 event_monitor_name

元素类型 信息

表 695. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

描述 创建事件数据流的事件监视器的名称。

用法 此元素允许您将要分析的数据与系统目录表中的特定事件监视器相关联。这是可在 SYSCAT.EVENTMONITORS 目录表的 NAME 列中找到的相同名称，该名称是在 CREATE EVENT MONITOR 和 SET EVENT MONITOR 语句上指定的。

partial_record － 部分记录

元素标识 partial_record

元素类型 信息

表 696. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表	event_table	-
表空间	event_tablespace	-
缓冲池	event_bufferpool	-
连接	event_conn	-
语句	event_stmt	-
语句	event_subsection	-
事务	event_xact	-

描述 指示事件监视器记录只是部分记录。

用法 在数据库释放之前，大多数事件监视器不会输出结果。可使用 FLUSH EVENT MONITORS 语句来强制将监视器值输出至事件监视器输出写程序。这允许您在不需要停止并重新启动事件监视器的情况下强制它将记录输出至写程序。此元素指示事件监视器记录是不是清空操作的结果并且因此成为部分记录。

清空事件监视器不会导致值复位。这意味着触发事件监视器时仍会生成完整的事件监视器记录。

event_time － 事件时间

元素标识 event_time

元素类型 信息

表 697. 事件监视信息

事件类型	逻辑数据分组	监视开关
表空间	event_tablespace	-
表	event_table	-

描述 发生事件的日期和时间。

用法 可使用此元素来帮助相关事件按时间排序。

evmon_flushes － 事件监视器清空数

元素标识 evmon_flushes

元素类型 信息

表 698. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表	event_table	-
表空间	event_tablespace	-

表 698. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
缓冲池	event_bufferpool	-

描述 发出 FLUSH EVENT MONITOR SQL 语句的次数。

用法 应用程序连接至数据库之后，数据库管理器每次成功处理 FLUSH EVENT MONITOR SQL 请求时，此标识就会递增。此元素有助于唯一标识数据库、表、表空间和缓冲池数据。

evmon_activates － 事件监视器激活数

元素标识 evmon_activates

元素类型 计数器

表 699. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表	event_table	-
表空间	event_tablespace	-
缓冲池	event_bufferpool	-
死锁	event_deadlock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

描述 激活事件监视器的次数。

用法 使用此元素以使上述事件类型返回的信息相关。此元素仅适用于写至表事件监视器。未对写至文件或管道的事件监视器保留此监视元素。

只有某些类型的写至表事件监视器使用 evmon_activates 监视元素（使用此元素的事件监视器类型列示在先前的表“事件监视信息”中）。在激活时，这些事件监视器将更新 SYSCAT.EVENTMONITORS 目录表的 evmon_activates 列。将记录此更改，所以 DATABASE CONFIGURATION 将显示:

```
数据库是一致的                                  = 否
```

如果事件监视器是使用 AUTOSTART 选项创建的，并且第一个用户连接至数据库后立即断开连接从而使得数据库被释放，则会生成日志文件。

sql_req_id － SQL 语句的请求标识

元素标识 sql_req_id

元素类型 信息

表 700. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-

描述 SQL 语句中某个操作的请求标识。

用法 第一个应用程序连接至数据库之后，数据库管理器每次成功处理 SQL 操作时，此标识就会递增。它的值在数据库中是唯一的，可以唯一地标识语句操作。

message － 控制表消息

元素标识 消息

元素类型 信息

表 701. 事件监视信息

事件类型	逻辑数据分组	监视开关
-	-	-

描述 MESSAGE_TIME 列中的时间戳记特征。此元素仅供写至表事件监视器在 CONTROL 表中使用。

用法 以下是可能的值:

FIRST_CONNECT
数据库激活后第一次连接至数据库的时间。

EVMON_START
EVMONNAME 列中列示的事件监视器启动的时间。

OVERFLOWS(*n*)
说明因为缓冲区溢出而删除了 *n* 个记录。

message_time － 时间戳记控制表消息

元素标识 message_time

元素类型 时间戳记

表 702. 事件监视信息

事件类型	逻辑数据分组	监视开关
-	-	-

描述 对应于 MESSAGE 列中描述的事件的时间戳记。此元素仅供写至表事件监视器在 CONTROL 表中使用。

partition_number － 分区号

元素标识 partition_number

元素类型 信息

表 703. 事件监视信息

事件类型	逻辑数据分组	监视开关
-	-	-

描述 此元素仅供分区数据库环境中的写至表事件监视器在目标 SQL 表中使用。此值指示插入事件监视器数据的分区的编号。

高可用性灾难恢复

高可用性灾难恢复监视元素

DB2 数据库高可用性灾难恢复（HADR）是一种数据库复制功能，它提供针对部分和整个站点故障的高可用性解决方案。

HADR 通过将数据更改从源数据库（称为主数据库）复制到目标数据库（称为备用数据库）来防止数据丢失。当主数据库发生故障时，可以很容易地将其转移至备用数据库。于是备用数据库成为新的主数据库。因为备用数据库服务器已经联机，所以可以非常迅速地完成故障转移，从而使停机时间缩至最短。

下列监视元素允许您检查 HADR 子系统的当前配置和状态。

- hadr_role － HADR 角色监视元素
- hadr_state － HADR 状态监视元素
- hadr_syncmode － HADR 同步方式监视元素
- hadr_connect_status － HADR 连接状态监视元素
- hadr_connect_time － HADR 连接时间监视元素
- hadr_heartbeat － HADR 脉动信号监视元素
- hadr_local_host － HADR 本地主机监视元素
- hadr_local_service － HADR 本地服务监视元素
- hadr_remote_host － HADR 远程主机监视元素
- hadr_remote_service － HADR 远程服务监视元素
- hadr_remote_instance － HADR 远程实例监视元素
- hadr_timeout － HADR 超时监视元素
- hadr_primary_log_file － HADR 主日志文件监视元素
- hadr_primary_log_page － HADR 主日志页监视元素
- hadr_primary_log_lsn － HADR 主日志 LSN监视元素
- hadr_standby_log_file － HADR 备用日志文件监视元素
- hadr_standby_log_page － HADR 备用日志页监视元素
- hadr_standby_log_lsn － HADR 备用日志 LSN 监视元素
- hadr_log_gap － HADR 日志间隔监视元素

相关概念:

- 『高可用性灾难恢复概述』（*《数据恢复及高可用性指南与参考》*）

hadr_role － HADR 角色

元素标识 hadr_role

元素类型 信息

表 704. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 数据库的当前 HADR 角色。此元素的数据类型是整型。此元素的值是下列其中一个常量:

- SQLM_HADR_ROLE_STANDARD: 数据库不是 HADR 数据库。
- SQLM_HADR_ROLE_PRIMARY: 数据库是主 HADR 数据库。
- SQLM_HADR_ROLE_STANDBY: 数据库是备用 HADR 数据库。

用法 使用此元素来确定数据库的 HADR 角色。

hadr_state － HADR 状态监视元素

元素标识 hadr_state

元素类型 信息

表 705. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 数据库的当前 HADR 状态。此元素的数据类型是整型。如果数据库的 HADR 角色是标准的，则应该忽略此元素。如果数据库具有 HADR 主角色或备用角色，则此元素的值是下列其中一个常量:

- SQLM_HADR_STATE_DISCONNECTED: 该数据库未连接至它的伙伴数据库。
- SQLM_HADR_STATE_LOC_CATCHUP: 该数据库正在执行本地同步复制。
- SQLM_HADR_STATE_REM_CATCH_PEND: 该数据库正在等待连接至它的伙伴数据库以执行远程同步复制。
- SQLM_HADR_STATE_REM_CATCHUP: 该数据库正在执行远程同步复制。
- SQLM_HADR_STATE_PEER: 在主数据库与备用数据库之间已建立连接，并且它们处于对等状态。

用法 使用此元素来确定数据库的 HADR 状态。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_syncmode － HADR 同步方式监视元素

元素标识 hadr_syncmode

元素类型 信息

表 706. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 数据库的当前 HADR 同步方式。此元素的数据类型是整型。如果数据库的

HADR 角色是标准的，则应该忽略此元素。如果数据库具有 HADR 主角色或备用角色，则此元素的值是下列其中一个常量:

- SQLM_HADR_SYNCMODE_SYNC: 同步方式。
- SQLM_HADR_SYNCMODE_NEARSYNC: 近似同步方式。
- SQLM_HADR_SYNCMODE_ASYNC: 异步方式。

用法 使用此元素来确定数据库的 HADR 同步方式。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_connect_status － HADR 连接状态监视元素

元素标识	hadr_connect_status
元素类型	信息

表 707. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 数据库的当前 HADR 连接状态。此元素的数据类型是整型。如果数据库的 HADR 角色是标准的，则应该忽略此元素。如果数据库具有 HADR 主角色或备用角色，则此元素的值是下列其中一个常量:

- SQLM_HADR_CONN_CONNECTED: 数据库连接至其伙伴节点。
- SQLM_HADR_CONN_DISCONNECTED: 数据库未连接至其伙伴节点。
- SQLM_HADR_CONN_CONGESTED: 数据库连接至其伙伴节点，但连接拥塞。当主数据库与备用数据库之间的 TCP/IP 套接字连接仍然活动但一端不能将信息发送至另一端时连接拥塞。例如，接收端未从套接字连接接收，导致 TCP/IP 发送空间变满。网络连接拥塞的原因包括下列几项:
  - 网络被太多资源共享，或者网络相对于主 HADR 节点的事务量而言不够快。
  - 备用 HADR 节点所在的服务器处理能力不足，无法以必要的速率检索通信子系统中的信息。

用法 使用此元素来确定数据库的 HADR 连接状态。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_connect_time － HADR 连接时间监视元素

元素标识	hadr_connect_time
元素类型	时间戳记

表 708. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 显示下列其中一项:

- HADR 连接时间
- HADR 拥塞时间
- HADR 断开连接时间

如果数据库的 HADR 角色是标准的，则应该忽略此元素。如果数据库为 HADR 主数据库或备用数据库，则此元素的含义取决于 *hadr_connect_status* 元素的值:

- 如果 *hadr_connect_status* 元素的值为 SQLM_HADR_CONN_CONNECTED，则此元素显示连接时间。
- 如果 *hadr_connect_status* 元素的值为 SQLM_HADR_CONN_CONGESTED，则此元素显示拥塞开始的时间。
- 如果 *hadr_connect_status* 元素的值为 SQLM_HADR_CONN_DISCONNECTED，则此元素显示断开连接时间。

如果自 HADR 引擎可拆离单元（EDU）启动后没有任何连接，则会将连接状态报告为“已断开连接”并且 HADR EDU 启动时间将用于断开连接时间。因为 HADR 连接和断开连接事件相对少见，所以即使 DFT_MON_TIMESTAMP 开关为 OFF，也会收集并报告该时间。

用法 使用此元素来确定当前 HADR 连接状态开始的时间。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 407 页的『hadr_connect_status － HADR 连接状态监视元素』
- 第 405 页的『hadr_role － HADR 角色』
- 『dft_monswitches － 缺省数据库系统监视器开关配置参数』（《性能指南》）

hadr_heartbeat － HADR 脉动信号监视元素

元素标识 hadr_heartbeat

元素类型 计数器

表 709. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

对于快照监视来说，无法复位此计数器。

描述 在 HADR 连接上已丢失的脉动信号数。如果数据库具有 HADR 主角色或备用角色，则此元素指示 HADR 连接的运行状况。脉动信号是以固定时间间隔从其他 HADR 数据库发送的消息。如果此元素的值为零，则表明未丢失脉动信号，并且连接的运行状况正常。此值越大，连接的运行状况就越差。

HADR 数据库期望伙伴数据库在特定时间间隔内至少发送一条脉动信号消息，该时间间隔是 HADR_TIMEOUT 数据库配置参数中定义的时间间隔的 1/4 或

者 30 秒（以较小者为准）。例如，如果 HADR_TIMEOUT 值是 80（秒），则 HADR 数据库期望伙伴数据库至少每 20 秒发送一条脉动信号消息。

注:

1. 此元素的数据类型是整型。
2. 如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定 HADR 连接的运行状况。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_local_host － HADR 本地主机监视元素

元素标识 hadr_local_host

元素类型 信息

表 710. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 本地 HADR 主机名。该值显示为主机名字符串或 IP 地址字符串，如“1.2.3.4”。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定有效 HADR 本地主机名称。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_local_service － HADR 本地服务监视元素

元素标识 hadr_local_service

元素类型 信息

表 711. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 本地 HADR TCP 服务。此值将显示为服务名称字符串或端口号字符串。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定有效 HADR 本地服务名称。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

• 第 405 页的『hadr_role – HADR 角色』

hadr_remote_host – HADR 远程主机监视元素

元素标识	hadr_remote_host
元素类型	信息

表 712. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 远程 HADR 主机名。该值显示为主机名字符串或 IP 地址字符串，如“1.2.3.4”。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定有效 HADR 远程主机名称。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

• 第 405 页的『hadr_role – HADR 角色』

hadr_remote_service – HADR 远程服务监视元素

元素标识	hadr_remote_service
元素类型	信息

表 713. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 远程 HADR TCP 服务。此值将显示为服务名称字符串或端口号字符串。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定有效 HADR 远程服务名称。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

• 第 405 页的『hadr_role – HADR 角色』

hadr_remote_instance – HADR 远程实例监视元素

元素标识	hadr_remote_instance
元素类型	信息

表 714. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 远程 HADR 实例名。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定有效 HADR 远程实例名。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_timeout － HADR 超时监视元素

元素标识 hadr_timeout

元素类型 信息

表 715. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 HADR 数据库服务器认为通信尝试失败所花的秒数。对于失败的尝试，HADR 数据库服务器在此元素列示的秒数内一定接收不到来自其伙伴的应答消息。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定有效 HADR 超时值。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_primary_log_file － HADR 主日志文件监视元素

元素标识 hadr_primary_log_file

元素类型 信息

表 716. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 当前日志文件在主 HADR 数据库上的名称。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定主 HADR 数据库上的当前日志文件。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_primary_log_page － HADR 主日志页监视元素

元素标识 hadr_primary_log_page

元素类型 信息

表 717. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 当前日志文件中的页号，指示当前日志在主 HADR 数据库上的位置。页号与日志文件相关。例如，页零是文件的开头。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定主 HADR 数据库上的当前日志页。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_primary_log_lsn － HADR 主日志 LSN 监视元素

元素标识 hadr_primary_log_lsn

元素类型 信息

表 718. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 主 HADR 数据库的当前日志位置。日志序号（LSN）是数据库的日志流中的字节位移。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定主 HADR 数据库上的当前日志位置。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role － HADR 角色』

hadr_standby_log_file － HADR 备用日志文件监视元素

元素标识 hadr_standby_log_file

元素类型 信息

表 719. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 当前日志文件在备用 HADR 数据库上的名称。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定备用 HADR 数据库上的当前日志文件。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role – HADR 角色』

hadr_standby_log_page – HADR 备用日志页监视元素

元素标识 hadr_standby_log_page

元素类型 信息

表 720. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 当前日志文件中的页号，指示当前日志在备用 HADR 数据库上的位置。页号与日志文件相关。例如，页零是文件的开头。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定备用 HADR 数据库上的当前日志页。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role – HADR 角色』

hadr_standby_log_lsn – HADR 备用日志 LSN 监视元素

元素标识 hadr_standby_log_lsn

元素类型 信息

表 721. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 备用 HADR 数据库的当前日志位置。日志序号（LSN）是数据库的日志流中的字节位移。如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定备用 HADR 数据库上的当前日志位置。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role – HADR 角色』

hadr_log_gap – HADR 日志间隔

元素标识 hadr_log_gap

元素类型 信息

表 722. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

描述 此元素显示主日志序号（LSN）与备用日志 LSN 之间正在运行的平均间隔。该间隔是以字节数度量的。

当日志文件被截断时，下一个日志文件中的 LSN 将会开始，就好像上一个文件未被截断一样。此 LSN 洞口未包含任何日志数据。这样的洞口可能导致日志间隔不返回主日志与备用日志之间的实际日志差别。

如果数据库的 HADR 角色是标准的，则应该忽略此元素。

用法 使用此元素来确定主 HADR 数据库日志与备用 HADR 数据库日志之间的间隔。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

相关参考:

- 第 405 页的『hadr_role – HADR 角色』

DB2 Connect

DB2 Connect 监视元素

下列元素提供数据库、应用程序、事务和语句级别的 DB2 连接信息:

- dcs_db_name – DCS 数据库名称监视元素
- host_db_name – 主机数据库名称监视元素
- gw_db_alias – 网关处的数据库别名监视元素
- gw_con_time – 启动的 DB2 Connect 网关首次连接监视元素
- gw_connections_top – 与主机数据库的最大并发连接数监视元素
- gw_total_cons – DB2 Connect 已尝试的总连接数监视元素
- gw_cur_cons – DB2 Connect 当前连接数监视元素
- gw_cons_wait_host – 正在等待主机应答的连接数监视元素
- gw_cons_wait_client – 正在等待客户机发送请求的连接数监视元素
- gw_exec_time – DB2 Connect 网关处理时间监视元素
- sql_stmts – 尝试的 SQL 语句数监视元素
- sql_chains – 尝试 SQL 链的次数监视元素sql_chains – 尝试 SQL 链的次数监视元素
- open_cursors – 打开的游标数监视元素
- dcs_appl_status – DCS 应用程序状态监视元素
- agent_status – DCS 应用程序代理程序数监视元素
- host_ccsid – 主机编码字符集标识监视元素

- outbound_comm_protocol – 出站通信协议监视元素
- outbound_comm_address – 出站通信地址监视元素
- inbound_comm_address – 入站通信地址监视元素
- inbound_bytes_received – 接收到的入站字节数监视元素
- outbound_bytes_sent – 发送的出站字节数监视元素
- outbound_bytes_received – 接收到的出站字节数监视元素
- inbound_bytes_sent – 发送的入站字节数监视元素
- outbound_bytes_sent_top – 发送的最大出站字节数监视元素
- outbound_bytes_received_top – 接收的最大出站字节数监视元素
- outbound_bytes_sent_bottom – 发送的最小出站字节数监视元素
- outbound_bytes_received_bottom – 接收的最小出站字节数监视元素
- max_data_sent_128 – 发送的出站字节数在 1 到 128 之间的语句数监视元素
- max_data_received_128 – 接收的出站字节数在 1 到 128 之间的语句数监视元素
- max_data_sent_256 – 发送的出站字节数在 129 到 256 之间的语句数监视元素
- max_data_received_256 – 接收的出站字节数在 129 到 256 之间的语句数监视元素
- max_data_sent_512 – 发送的出站字节数在 257 到 512 之间的语句数监视元素
- max_data_received_512 – 接收的出站字节数在 257 到 512 之间的语句数监视元素
- max_data_sent_1024 – 发送的出站字节数在 513 到 1024 之间的语句数监视元素
- max_data_received_1024 – 接收的出站字节数在 513 到 1024 之间的语句数监视元素
- max_data_sent_2048 – 发送的出站字节数在 1025 到 2048 之间的语句数监视元素
- max_data_received_2048 – 接收的出站字节数在 1025 到 2048 之间的语句数监视元素
- max_data_sent_4096 – 发送的出站字节数在 2049 到 4096 之间的语句数监视元素
- max_data_received_4096 – 接收的出站字节数在 2049 到 4096 之间的语句数监视元素
- max_data_sent_8192 – 发送的出站字节数在 4097 到 8192 之间的语句数监视元素
- max_data_received_8192 – 接收的出站字节数在 4097 到 8092 之间的语句数监视元素
- max_data_sent_16384 – 发送的出站字节数在 8193 到 16384 之间的语句数监视元素
- max_data_received_16384 – 接收的出站字节数在 8193 到 16384 之间的语句数监视元素
- max_data_sent_31999 – 发送的出站字节数在 16385 到 31999 之间的语句数监视元素
- max_data_received_31999 – 接收的出站字节数在 16385 到 31999 之间的语句数监视元素
- max_data_sent_64000 – 发送的出站字节数在 32000 到 64000 之间的语句数监视元素
- max_data_received_64000 – 接收的出站字节数在 32000 到 64000 之间的语句数监视元素

- max_data_sent_gt64000 – 发送的出站字节数大于 64000 的语句数监视元素
- max_data_received_gt64000 – 接收的出站字节数大于 64000 的语句数监视元素
- max_network_time_1_ms – 网络时间最多为 1 ms 的语句数监视元素
max_network_time_1_ms – 网络时间最多为 1 ms 的语句数监视元素
- max_network_time_4_ms – 网络时间在 1 到 4 ms 之间的语句数监视元素
- max_network_time_16_ms – 网络时间在 4 到 16 ms 之间的语句数监视元素
- max_network_time_100_ms – 网络时间在 16 到 100 ms 之间的语句数监视元素
max_network_time_100_ms – 网络时间在 16 到 100 ms 之间的语句数监视元素
- max_network_time_500_ms – 网络时间在 100 到 500 ms 之间的语句数监视元素
max_network_time_500_ms – 网络时间在 100 到 500 ms 之间的语句数监视元素
- max_network_time_gt500_ms – 网络时间在 500 ms 以上的语句数监视元素
max_network_time_gt500_ms – 网络时间在 500 ms 以上的语句数监视元素
- network_time_top – 语句的最大网络时间监视元素
- network_time_bottom – 语句的最小网络时间监视元素
- xid – 事务标识监视元素
- elapsed_exec_time – 执行语句耗用时间监视元素
- host_response_time – 主机响应时间监视元素
- num_transmissions – 传输次数监视元素
- num_transmissions_group – 传输组数监视元素num_transmissions_group – 传输组数监视元素
- con_response_time – 最近的连接响应时间监视元素
- con_elapsed_time – 最近的连接耗用时间监视元素
- gw_comm_errors – 通信错误监视元素
- gw_comm_error_time – 通信错误时间监视元素
- blocking_cursor – 分块游标监视元素
- 事务处理器监视监视元素

dcs_db_name – DCS 数据库名称

元素标识 dcs_db_name

元素类型 信息

表 723. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

描述 DCS 目录中编目的 DCS 数据库的名称。

用法 此元素用于 DCS 应用程序的问题确定。

相关参考:

- 第 417 页的『host_db_name – 主机数据库名称』
- 第 417 页的『gw_db_alias – 网关上的数据库别名』

host_db_name – 主机数据库名称

元素标识 host_db_name

元素类型 信息

表 724. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

描述 对其收集信息或应用程序连接至的主机数据库的真实名称。这是创建数据库时给定的名称。

用法 此元素用于 DCS 应用程序的问题确定。

相关参考:

- 第 416 页的『dcs_db_name – DCS 数据库名称』
- 第 417 页的『gw_db_alias – 网关上的数据库别名』

gw_db_alias – 网关上的数据库别名

元素标识 gw_db_alias

元素类型 信息

表 725. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

描述 DB2 Connect 网关上用来连接至主机数据库的别名。

用法 此元素用于 DCS 应用程序的问题确定。

相关参考:

- 第 416 页的『dcs_db_name – DCS 数据库名称』
- 第 417 页的『host_db_name – 主机数据库名称』

gw_con_time – DB2 Connect 网关首次启动的连接

元素标识 gw_con_time

元素类型 时间戳记

表 726. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记
DCS 应用程序	dcs_appl	时间戳记

描述 从 DB2 Connect 网关首次启动与主机数据库的连接的日期和时间。

用法 此元素用于 DCS 应用程序的问题确定。

gw_connections_top － 与主机数据库的最大并行连接数

元素标识 gw_connections_top

元素类型 水位标记

表 727. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本

描述 自首次数据库连接后由 DB2 Connect 网关处理的与主机数据库的并行连接的最大数目。

用法 此元素帮助您了解 DB2 Connect 网关上的活动级别及系统资源的关联使用。

相关参考:

- 第 418 页的『gw_total_cons － 对 DB2 Connect 尝试连接的总数』
- 第 418 页的『gw_cur_cons － DB2 Connect 的当前连接数』

gw_total_cons － 对 DB2 Connect 尝试连接的总数

元素标识 gw_total_cons

元素类型 水位标记

表 728. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

可将快照监视的计数器复位。

描述 自最后一次 `db2start` 命令或最后一次复位后尝试从 DB2 Connect 网关进行连接的总次数。

用法 此元素帮助您了解 DB2 Connect 网关上的活动级别及系统资源的关联使用。

相关参考:

- 第 418 页的『gw_connections_top － 与主机数据库的最大并行连接数』
- 第 418 页的『gw_cur_cons － DB2 Connect 的当前连接数』

gw_cur_cons － DB2 Connect 的当前连接数

元素标识 gw_cur_cons

元素类型 标尺

表 729. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

描述 由 DB2 Connect 网关处理的与主机数据库的当前连接的数目。

用法 此元素帮助您了解 DB2 Connect 网关上的活动级别及系统资源的关联使用。

相关参考:

- 第 418 页的『gw_connections_top – 与主机数据库的最大并行连接数』
- 第 418 页的『gw_total_cons – 对 DB2 Connect 尝试连接的总数』

gw_cons_wait_host – 等待主机应答的连接数

元素标识 gw_cons_wait_host

元素类型 标尺

表 730. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

描述 由 DB2 Connect 网关处理并且等待主机应答的与主机数据库的当前连接的数目。

用法 此值可能经常更改。在延长时间段内应按一定时间间隔对其进行采样以了解真实的网关使用情况。

相关参考:

- 第 418 页的『gw_cur_cons – DB2 Connect 的当前连接数』
- 第 419 页的『gw_cons_wait_client – 等待客户机发送请求的连接数』

gw_cons_wait_client – 等待客户机发送请求的连接数

元素标识 gw_cons_wait_client

元素类型 标尺

表 731. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

描述 由 DB2 Connect 网关处理并且等待客户机发送请求的与主机数据库的当前连接的数目。

用法 此值可能经常更改。在延长时间段内应按一定时间间隔对其进行采样以了解真实的网关使用情况。

相关参考:

- 第 418 页的『gw_cur_cons – DB2 Connect 的当前连接数』
- 第 419 页的『gw_cons_wait_host – 等待主机应答的连接数』

gw_exec_time – DB2 Connect 网关处理所耗用的时间

元素标识 gw_exec_time

元素类型 时间

表 732. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

可将快照监视的计数器复位。

描述 DB2 Connect 网关处理应用程序请求（自建立连接后）或处理单个语句所花的时间（以秒和微秒计）。

用法 使用此元素来确定整体处理时间的哪一部分用于 DB2 Connect 网关处理。

sql_stmts － 尝试的 SQL 语句数

元素标识 sql_stmts

元素类型 计数器

表 733. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本
数据传输	stmt_transmissions	基本

可将快照监视的计数器复位。

描述 对于数据传输快照，此元素表示语句处理期间在 DB2 Connect 网关与主机之间采用 *n* 个数据传输的 SQL 语句数。范围 *n* 是由 *num_transmissions_group* 元素指定的。

对于 DCS DATABASE 快照，此语句计数为自激活数据库后处理的语句数。

对于 DCS APPLICATION 快照，此语句计数为自此应用程序建立与数据库的连接后处理的语句数。

用法 使用此元素来度量数据库或应用程序级别的数据库活动。要计算给定时间段的 SQL 语句吞吐量，可按两次快照之间所耗用的时间来分割此元素。

对于数据传输级别：使用此元素来获取有关处理期间使用 2、3、4（等等）个数据传输的语句数的统计信息。（要处理语句，至少需要 2 个数据传输：发送和接收。）这些统计信息能够让您更好地了解数据库或应用程序级别的数据库或应用程序活动和网络流量。

注:

1. *sql_stmts* 监视元素表示尝试将 SQL 语句发送至服务器的次数:
   - 在应用程序级别和数据库级别，游标内的每个 SQL 语句是分开计数的。
   - 在传输级别，同一游标内的所有语句在计数时都被当作单个 SQL 语句。

相关参考:

- 第 444 页的『num_transmissions_group － 传输次数组』
- 第 421 页的『sql_chains － 尝试的 SQL 链数』
- 第 398 页的『time_stamp － 快照时间』

sql_chains － 尝试的 SQL 链数

元素标识 sql_chains

元素类型 计数器

表 734. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	基本

可将快照监视的计数器复位。

描述 表示语句处理期间在 DB2 Connect 网关与主机之间采用 *n* 个数据传输的 SQL 语句数。范围 *n* 是由 *num_transmissions_group* 元素指定的。

例如，如果链接已打开，并且 PREP 和 OPEN 语句链接在一起，该链一共占用了两个传输，则 *sql_chains* 报告为“1”，而 *sql_stmts* 报告为“2”。

如果链接已关闭，则 *sql_chains* 计数等于 *sql_stmts* 计数。

用法 使用此元素来获取有关处理期间使用 2、3、4（等等）个数据传输的语句数的统计信息。（要处理语句，至少需要 2 个数据传输：发送和接收。）这些统计信息能够让您更好地了解数据库或应用程序级别的数据库或应用程序活动和网络流量。

注： *sql_stmts* 监视元素表示尝试将 SQL 语句发送至服务器的次数。在传输级别，同一游标内的所有语句在计数时都被当作单个 SQL 语句。

相关参考：

- 第 444 页的『num_transmissions_group － 传输次数组』
- 第 420 页的『sql_stmts － 尝试的 SQL 语句数』
- 『Statement attributes (CLI) list』（*Call Level Interface Guide and Reference, Volume 2*）

open_cursors － 打开的游标数

元素标识 open_cursors

元素类型 标尺

表 735. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句

描述 当前对应用程序打开的游标数。

用法 使用此元素来评估分配的内存量。DB2 客户机、DB2 Connect 或目标数据库上的数据库代理分配的内存量与当前打开的游标数有关。了解此信息可帮助您进行容量规划。例如，每个执行分块的打开游标的缓冲区大小为 RQRIOBLK。如果启用了 *deferred_prepare*，则将分配两个缓冲区。

此元素不包括之前的关闭操作关闭的游标。之前的关闭操作是在主机数据库将最后一个记录返回至客户机时发生的。游标在主机和网关上关闭，但在客户机上仍然是打开的。可使用 DB2 调用级接口来设置之前的关闭游标。

dcs_appl_status – DCS 应用程序状态

元素标识 dcs_appl_status

元素类型 信息

表 736. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

描述 DB2 Connect 网关上的 DCS 应用程序的状态。

用法 此元素用于 DCS 应用程序的问题确定。值包括:

- SQLM_DCS_CONNECTPEND_OUTBOUND

  应用程序已启动从 DB2 Connect 网关至主机数据库的数据库连接，但请求尚未完成。
- SQLM_DCS_UOWWAIT_OUTBOUND

  DB2 Connect 网关正在等待主机数据库应答应用程序的请求。
- SQLM_DCS_UOWWAIT_INBOUND

  已建立从 DB2 Connect 网关至主机数据库的连接并且网关正在等待来自应用程序的 SQL 请求。或者 DB2 Connect 网关正在代表应用程序中的工作单元等待。这通常意味着正在执行应用程序的代码。

相关参考:

- 第 423 页的『host_ccsid – 主机编码字符集标识』
- 第 423 页的『outbound_comm_protocol – 出站通信协议』
- 第 424 页的『outbound_comm_address – 出站通信地址』
- 第 424 页的『inbound_comm_address – 入站通信地址』

agent_status – DCS 应用程序代理程序数

元素标识 agent_status

元素类型 信息

表 737. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

描述 在连接集中器环境中，此值显示当前具有关联代理程序的应用程序。

用法 值包括:

- SQLM_AGENT_ASSOCIATED

  代表此应用程序工作的代理程序与其相关联。
- SQLM_AGENT_NOT_ASSOCIATED

代表此应用程序工作的代理程序不再与其相关联并且正被另一应用程序使用。没有关联代理程序的应用程序下一次完成工作时，将重新关联代理程序。

相关参考:

- 第 422 页的『dcs_appl_status － DCS 应用程序状态』

host_ccsid － 主机编码字符集标识

元素标识 host_ccsid

元素类型 信息

表 738. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

描述 此项是主机数据库的编码字符集标识（CCSID）。

用法 此元素用于 DCS 应用程序的问题确定。

相关参考:

- 第 422 页的『dcs_appl_status － DCS 应用程序状态』
- 第 423 页的『outbound_comm_protocol － 出站通信协议』
- 第 424 页的『outbound_comm_address － 出站通信地址』
- 第 424 页的『inbound_comm_address － 入站通信地址』

outbound_comm_protocol － 出站通信协议

元素标识 outbound_comm_protocol

元素类型 信息

表 739. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

描述 DB2 Connect 网关与主机之间使用的通信协议。

用法 此元素用于 DCS 应用程序的问题确定。有效值为:
- SQLM_PROT_APPC
- SQLM_PROT_TCPIP

相关参考:

- 第 422 页的『dcs_appl_status － DCS 应用程序状态』
- 第 423 页的『host_ccsid － 主机编码字符集标识』
- 第 424 页的『outbound_comm_address － 出站通信地址』
- 第 424 页的『inbound_comm_address － 入站通信地址』

outbound_comm_address － 出站通信地址

元素标识 outbound_comm_address

元素类型 信息

表 740. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl_info	基本

描述 此项是目标数据库的通信地址。例如，它可能是 SNA 网络标识和 LU 伙伴名称，或者是 TCP/IP 的 IP 地址和端口号。

用法 此元素用于 DCS 应用程序的问题确定。

相关参考:

- 第 422 页的『dcs_appl_status － DCS 应用程序状态』
- 第 423 页的『host_ccsid － 主机编码字符集标识』
- 第 423 页的『outbound_comm_protocol － 出站通信协议』
- 第 424 页的『inbound_comm_address － 入站通信地址』

inbound_comm_address － 入站通信地址

元素标识 inbound_comm_address

元素类型 信息

表 741. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl_info	基本

描述 此项是客户机的通信地址。例如，它可能是 SNA 网络标识和 LU 伙伴名称，或者是 TCP/IP 的 IP 地址和端口号。

用法 此元素用于 DCS 应用程序的问题确定。

相关参考:

- 第 422 页的『dcs_appl_status － DCS 应用程序状态』
- 第 423 页的『host_ccsid － 主机编码字符集标识』
- 第 423 页的『outbound_comm_protocol － 出站通信协议』
- 第 424 页的『outbound_comm_address － 出站通信地址』

inbound_bytes_received － 接收的入站字节数

元素标识 inbound_bytes_received

元素类型 计数器

表 742. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句

对于应用程序级别的快照监视，可复位此计数器。不能在其他级别复位此计数器。

描述 DB2 Connect 网关从客户机接收的字节数，通信协议开销（如 TCP/IP 或 SNA 头）除外。

用法 使用此元素来度量从客户机至 DB2 Connect 网关的吞吐量。

相关参考:

- 第 425 页的『outbound_bytes_sent – 发送的出站字节数』
- 第 425 页的『outbound_bytes_received – 接收的出站字节数』
- 第 426 页的『inbound_bytes_sent – 发送的入站字节数』

outbound_bytes_sent – 发送的出站字节数

元素标识 outbound_bytes_sent

元素类型 计数器

表 743. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句
数据传输	stmt_transmissions	语句

对于语句级别的快照监视，不能复位此计数器。可在其他级别复位此计数器。

描述 DB2 Connect 网关发送至主机的字节数，通信协议开销（如 TCP/IP 或 SNA 头）除外。

对于数据传输级别：处理使用此数目的数据传输的所有语句期间 DB2 Connect 网关发送至主机的字节数。

用法 使用此元素来度量从 DB2 Connect 网关至主机数据库的吞吐量。

相关参考:

- 第 424 页的『inbound_bytes_received – 接收的入站字节数』
- 第 425 页的『outbound_bytes_received – 接收的出站字节数』
- 第 426 页的『inbound_bytes_sent – 发送的入站字节数』

outbound_bytes_received – 接收的出站字节数

元素标识 outbound_bytes_received

元素类型 计数器

表 744. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句
数据传输	stmt_transmissions	语句

对于语句级别的快照监视，不能复位此计数器。可在其他级别复位此计数器。

描述 DB2 Connect 网关从主机接收的字节数，通信协议开销（如 TCP/IP 或 SNA 头）除外。

对于数据传输级别：处理使用此数目的数据传输的所有语句期间 DB2 Connect 网关从主机接收的字节数。

用法 使用此元素来度量从主机数据库至 DB2 Connect 网关的吞吐量。

相关参考:

- 第 424 页的『inbound_bytes_received – 接收的入站字节数』
- 第 425 页的『outbound_bytes_sent – 发送的出站字节数』
- 第 426 页的『inbound_bytes_sent – 发送的入站字节数』

inbound_bytes_sent – 发送的入站字节数

元素标识 inbound_bytes_sent

元素类型 计数器

表 745. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句

对于应用程序级别的快照监视，可复位此计数器。不能在其他级别复位此计数器。

描述 DB2 Connect 网关发送至客户机的字节数，通信协议开销（如 TCP/IP 或 SNA 头）除外。

用法 使用此元素来度量从 DB2 Connect 网关至客户机的吞吐量。

相关参考:

- 第 424 页的『inbound_bytes_received – 接收的入站字节数』
- 第 425 页的『outbound_bytes_sent – 发送的出站字节数』
- 第 425 页的『outbound_bytes_received – 接收的出站字节数』

outbound_bytes_sent_top – 发送的最大出站字节数

元素标识 outbound_bytes_sent_top

元素类型 水位标记

表 746. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

描述 在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链发送至主机的最多字节数。

用法 将此元素与“发送的出站字节数”一起使用，将后者作为说明 DB2 Connect 网关至主机数据库的吞吐量的另一参数。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

outbound_bytes_received_top – 接收的最大出站字节数

元素标识	outbound_bytes_received_top
元素类型	水位标记

表 747. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

描述 在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链从主机接收的最多字节数。

用法 将此元素与“接收的出站字节数”一起使用，将后者作为说明主机数据库至 DB2 Connect 网关的吞吐量的另一参数。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

outbound_bytes_sent_bottom – 发送的最小出站字节数

元素标识	outbound_bytes_sent_bottom
元素类型	水位标记

表 748. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

描述 在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链发送至主机的最少字节数。

用法 将此元素与“发送的出站字节数”一起使用，将后者作为说明 DB2 Connect 网关至主机数据库的吞吐量的另一参数。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』

- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

outbound_bytes_received_bottom – 接收的最小出站字节数

元素标识 outbound_bytes_received_bottom

元素类型 水位标记

表 749. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

描述 在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链从主机接收的最少字节数。

用法 将此元素与“接收的出站字节数”一起使用，将后者作为说明主机数据库至 DB2 Connect 网关的吞吐量的另一参数。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_sent_128 – 发送的出站字节数在 1 到 128 字节之间的语句数

元素标识 max_data_sent_128

元素类型 计数器

表 750. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 1 到 128 字节之间（包括 1 字节和 128 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_received_128 – 接收的出站字节数在 1 到 128 字节之间的语句数

元素标识 max_data_received_128

元素类型 计数器

表 751. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 1 到 128 字节之间（包括 1 字节和 128 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_sent_256 – 发送的出站字节数在 129 到 256 字节之间的语句数

元素标识 max_data_sent_256

元素类型 计数器

表 752. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 129 到 256 字节之间（包括 129 字节和 256 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_received_256 – 接收的出站字节数在 129 到 256 字节之间的语句数

元素标识 max_data_received_256

元素类型 计数器

表 753. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句

表 753. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 129 到 256 字节之间（包括 129 字节和 256 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains － 尝试的 SQL 链数』
- 第 420 页的『sql_stmts － 尝试的 SQL 语句数』

max_data_sent_512 － 发送的出站字节数在 257 到 512 字节之间的语句数

元素标识	max_data_sent_512
元素类型	计数器

表 754. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 257 到 512 字节之间（包括 257 字节和 512 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains － 尝试的 SQL 链数』
- 第 420 页的『sql_stmts － 尝试的 SQL 语句数』

max_data_received_512 － 接收的出站字节数在 257 到 512 字节之间的语句数

元素标识	max_data_received_512
元素类型	计数器

表 755. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 257 到 512 字节之间（包括 257 字节和 512 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_sent_1024 – 发送的出站字节数在 513 到 1024 字节之间的语句数

元素标识	max_data_sent_1024
元素类型	计数器

表 756. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 513 到 1024 字节之间（包括 513 字节和 1024 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_received_1024 – 接收的出站字节数在 513 到 1024 字节之间的语句数

元素标识	max_data_received_1024
元素类型	计数器

表 757. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 513 到 1024 字节之间（包括 513 字节和 1024 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:
- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_sent_2048 – 发送的出站字节数在 1025 到 2048 字节之间的语句数

元素标识 max_data_sent_2048

元素类型 计数器

表 758. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 1025 到 2048 字节之间（包括 1025 字节和 2048 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:
- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_received_2048 – 发送的出站字节数在 1025 到 2048 字节之间的语句数

元素标识 max_data_received_2048

元素类型 计数器

表 759. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 1025 到 2048 字节之间（包括 1025 字节和 2048 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:
- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_sent_4096 – 发送的出站字节数在 2049 到 4096 字节之间的语句数

元素标识 max_data_sent_4096

元素类型 计数器

表 760. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 2049 到 4096 字节之间（包括 2049 字节和 4096 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_received_4096 – 接收的出站字节数在 2049 到 4096 字节之间的语句数

元素标识 max_data_received_4096

元素类型 计数器

表 761. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 2049 到 4096 字节之间（包括 2049 字节和 4096 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_sent_8192 – 发送的出站字节数在 4097 到 8192 字节之间的语句数

元素标识 max_data_sent_8192

元素类型 计数器

表 762. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 4097 到 8192 字节之间（包括 4097 字节和 8192 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_received_8192 – 接收的出站字节数在 4097 到 8192 字节之间的语句数

元素标识 max_data_received_8192

元素类型 计数器

表 763. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 4097 到 8192 字节之间（包括 4097 字节和 8192 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_sent_16384 – 发送的出站字节数在 8193 到 16384 字节之间的语句数

元素标识 max_data_sent_16384

元素类型 计数器

表 764. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 8193 到 16384 字节之间（包括 8193 字节和 16384 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains － 尝试的 SQL 链数』
- 第 420 页的『sql_stmts － 尝试的 SQL 语句数』

max_data_received_16384 － 发送的出站字节数在 8193 到 16384 字节之间的语句数

元素标识	max_data_received_16384
元素类型	计数器

表 765. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 8193 到 16384 字节之间（包括 8193 字节和 16384 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains － 尝试的 SQL 链数』
- 第 420 页的『sql_stmts － 尝试的 SQL 语句数』

max_data_sent_31999 － 发送的出站字节数在 16385 到 31999 字节之间的语句数

元素标识	max_data_sent_31999
元素类型	计数器

表 766. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句

表 766. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 16385 到 31999 字节之间（包括 16385 字节和 31999 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_received_31999 – 接收的出站字节数在 16385 到 31999 字节之间的语句数

元素标识 max_data_received_31999

元素类型 计数器

表 767. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 16385 到 31999 字节之间（包括 16385 字节和 31999 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_sent_64000 – 发送的出站字节数在 32000 到 64000 字节之间的语句数

元素标识 max_data_sent_64000

元素类型 计数器

表 768. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数在 32000 到 64000 字节之间（包括 32000 字节和 64000 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_received_64000 – 接收的出站字节数在 32000 到 64000 字节之间的语句数

元素标识 max_data_received_64000

元素类型 计数器

表 769. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数在 32000 到 64000 字节之间（包括 32000 字节和 64000 字节）的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_sent_gt64000 – 发送的出站字节数高于 64000 的语句数

元素标识 max_data_sent_gt64000

元素类型 计数器

表 770. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示发送的出站字节数高于 64000 的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_data_received_gt64000 – 接收的出站字节数高于 64000 的语句数

元素标识 max_data_received_gt64000

元素类型 计数器

表 771. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示接收的出站字节数高于 64000 的语句或链的数目。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』

max_network_time_1_ms – 网络时间最多为 1 ms 的语句数

元素标识 max_network_time_1_ms

元素类型 计数器

表 772. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示网络时间小于或等于 1 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 443 页的『host_response_time – 主机响应时间』
- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』
- 第 389 页的『total_exec_time – 执行语句所耗用的时间』

max_network_time_4_ms – 网络时间在 1 到 4 ms 之间的语句数

元素标识 max_network_time_4_ms

元素类型 计数器

表 773. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示网络时间大于 1 毫秒但小于或等于 4 毫秒的语句或链的数目。(网络时间是主机响应时间与语句或链所耗用的执行时间之差。)

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 443 页的『host_response_time – 主机响应时间』
- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』
- 第 389 页的『total_exec_time – 执行语句所耗用的时间』

max_network_time_16_ms – 网络时间在 4 到 16 ms 之间的语句数

元素标识 max_network_time_16_ms

元素类型 计数器

表 774. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示网络时间大于 4 毫秒但小于或等于 16 毫秒的语句或链的数目。(网络时间是主机响应时间与语句或链所耗用的执行时间之差。)

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 443 页的『host_response_time – 主机响应时间』
- 第 421 页的『sql_chains – 尝试的 SQL 链数』
- 第 420 页的『sql_stmts – 尝试的 SQL 语句数』
- 第 389 页的『total_exec_time – 执行语句所耗用的时间』

max_network_time_100_ms － 网络时间在 16 到 100 ms 之间的语句数

元素标识 max_network_time_100_ms

元素类型 计数器

表 775. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示网络时间大于 16 毫秒但小于或等于 100 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 443 页的『host_response_time － 主机响应时间』
- 第 421 页的『sql_chains － 尝试的 SQL 链数』
- 第 420 页的『sql_stmts － 尝试的 SQL 语句数』
- 第 389 页的『total_exec_time － 执行语句所耗用的时间』

max_network_time_500_ms － 网络时间在 100 到 500 ms 之间的语句数

元素标识 max_network_time_500_ms

元素类型 计数器

表 776. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示网络时间大于 100 毫秒但小于或等于 500 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 443 页的『host_response_time － 主机响应时间』
- 第 421 页的『sql_chains － 尝试的 SQL 链数』
- 第 420 页的『sql_stmts － 尝试的 SQL 语句数』
- 第 389 页的『total_exec_time － 执行语句所耗用的时间』

max_network_time_gt500_ms － 网络时间大于 500 ms 的语句数

元素标识 max_network_time_gt500_ms

元素类型 计数器

表 777. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

描述 此元素表示网络时间大于 500 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 443 页的『host_response_time － 主机响应时间』
- 第 421 页的『sql_chains － 尝试的 SQL 链数』
- 第 420 页的『sql_stmts － 尝试的 SQL 语句数』
- 第 389 页的『total_exec_time － 执行语句所耗用的时间』

network_time_top － 语句的最长网络时间

元素标识 network_time_top

元素类型 水位标记

表 778. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句，时间戳记
DCS 应用程序	dcs_appl	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

可将快照监视的计数器复位。

描述 此元素表示对此 DCS 数据库或在此 DCS 应用程序中执行语句的最长网络时间，或者表示执行使用这么多数据传输的语句的最长网络时间。（网络时间是主机响应时间与语句所耗用的执行时间之差。）

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。注意，时间戳记开关设置为 OFF 时不收集此元素。

相关参考:

- 第 389 页的『total_exec_time － 执行语句所耗用的时间』
- 第 443 页的『host_response_time － 主机响应时间』

network_time_bottom － 语句的最短网络时间

元素标识 network_time_bottom

元素类型 水位标记

表 779. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句，时间戳记
DCS 应用程序	dcs_appl	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

可将快照监视的计数器复位。

描述 此元素表示对此 DCS 数据库或在此 DCS 应用程序中执行语句的最短网络时间，或者表示执行使用这么多数据传输的语句的最短网络时间。(网络时间是主机响应时间与语句所耗用的执行时间之差。)

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

相关参考:

- 第 389 页的『total_exec_time － 执行语句所耗用的时间』
- 第 443 页的『host_response_time － 主机响应时间』

xid － 事务标识

元素标识 xid

元素类型 信息

表 780. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	工作单元

描述 事务管理器在两阶段落实事务中生成的唯一事务标识(在所有数据库上)。

用法 此标识可用来将事务管理器生成的事务与对多个数据库执行的事务相关联。它还可用来帮助诊断事务管理器问题，方法是将涉及两阶段落实协议的数据库事务与事务管理器生成的事务关联在一起。

elapsed_exec_time － 语句执行耗用时间

元素标识 elapsed_exec_time

元素类型 时间

表 781. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句，时间戳记
应用程序	appl	语句，时间戳记
DCS 数据库	dcs_dbase	语句，时间戳记
DCS 应用程序	dcs_appl	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

表 781. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句，时间戳记

对于语句级别的快照监视，不能复位此计数器。可在其他级别复位此计数器。

描述 在 DCS 语句级别，这是在主机数据库服务器上处理 SQL 请求所耗用的时间。此值由此服务器报告。和 host_response_time 元素相比，此元素不包括 DB2 Connect 与主机数据库服务器之间的网络耗用时间。

在其他级别，此值表示对特定数据库或应用程序执行的所有语句的主机执行时间的总和，或者是使用给定数目的数据传输的那些语句的主机执行时间的总和。

用法 将此元素与其他耗用时间监视元素一起使用以评估数据库服务器的 SQL 请求处理情况从而帮助隔离性能问题。

从 host_response_time 元素减去此元素，以计算 DB2 Connect 与主机数据库服务器之间的网络耗用时间。

注: 对于 dcs_dbase、dcs_appl、dcs_stmt 和 stmt_transmissions 级别，*elapsed_exec_time* 元素仅适用于 z/OS 数据库。如果 DB2 Connect 网关连接至 Windows、Linux、AIX 或其他 UNIX 数据库，则 *elapsed_exec_time* 报告为零。

相关参考:

- 第 443 页的『host_response_time － 主机响应时间』

host_response_time － 主机响应时间

元素标识 host_response_time

元素类型 时间

表 782. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

对于语句级别的快照监视，不能复位此计数器。可在其他级别复位此计数器。

描述 在 DCS 语句级别，此项是语句从 DB2 Connect 网关发送至主机以进行处理的时间与从主机接收到结果的时间之间所耗用的时间。在 DCS 数据库和 DCS 应用程序级别，此项是对特定应用程序或数据库执行的所有语句所耗用的时间。在数据传输级别，此项是使用这么多数据传输的所有语句的主机响应时间的总和。

用法 将此元素与发送的出站字节数和接收的出站字节数配合使用来计算出站响应时间（传输速率）:

```
(发送的出站字节数 + 接收的出站字节数) / 主机响应时间
```

相关参考:

• 第 425 页的『outbound_bytes_received – 接收的出站字节数』
• 第 425 页的『outbound_bytes_sent – 发送的出站字节数』
• 第 442 页的『elapsed_exec_time – 语句执行耗用时间』

num_transmissions – 传输次数

元素标识 num_transmissions

元素类型 计数器

表 783. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 语句	dcs_stmt	语句

描述

这是旧的监视元素，DB2 UDB 版本 8.1.2 或更高版本中已不再使用此元素。如果在使用 DB2 UDB 版本 8.1.2 或更高版本，则参阅 num_transmissions_group 监视元素。

DB2 Connect 网关与用于处理此 DCS 语句的主机之间的数据传输次数。（一次数据传输包括一次发送或一次接收）。

用法 使用此元素来更好地了解特定语句执行时间很长的原因。例如，返回的结果集很大的查询可能需要许多次数据传输才能完成。

相关参考:

• 第 444 页的『num_transmissions_group – 传输次数组』

num_transmissions_group – 传输次数组

元素标识 num_transmissions_group

元素类型 信息

表 784. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 语句	dcs_stmt	语句

描述 DB2 Connect 网关与用于处理此 DCS 语句的主机之间的数据传输范围。（一次数据传输包括一次发送或一次接收）。

用法 使用此元素来更好地了解特定语句执行时间很长的原因。例如，返回的结果集很大的查询可能需要许多次数据传输才能完成。

下面描述了表示传输范围的常量，它们是在 sqlmon.h 中定义的。

API 常量	描述
SQLM_DCS_TRANS_GROUP_2	2 次传输
SQLM_DCS_TRANS_GROUP_3TO7	3 到 7 次传输
SQLM_DCS_TRANS_GROUP_8TO15	8 到 15 次传输
SQLM_DCS_TRANS_GROUP_16TO64	16 到 64 次传输
SQLM_DCS_TRANS_GROUP_GT64	大于 64 次传输

con_response_time － 连接的最新响应时间

元素标识 con_response_time

元素类型 时间

表 785. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记

描述 对于连接至此数据库的最新 DCS 应用程序，此项为连接处理开始与实际建立连接之间所耗用的时间。

用法 将此元素用作当前应用程序连接至特定主机数据库所花时间的指示符。

相关参考:

- 第 265 页的『pkg_cache_num_overflows － 程序包高速缓存溢出数』

con_elapsed_time － 最新连接耗用时间

元素标识 con_elapsed_time

元素类型 时间

表 786. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记

描述 连接最新与此主机数据库断开连接的 DCS 应用程序所耗用的时间。

用法 将此元素用作应用程序保持与主机数据库的连接的时间长度的指示符。

相关参考:

- 第 265 页的『pkg_cache_num_overflows － 程序包高速缓存溢出数』

gw_comm_errors － 通信错误数

元素标识 gw_comm_errors

元素类型 计数器

表 787. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本

可将快照监视的计数器复位。

描述 DCS 应用程序尝试连接至主机数据库时或处理 SQL 语句时发生通信错误（SQL30081）的次数。

用法 通过监视一段时间内发生通信错误的次数，可评估 DB2 Connect 网关与特定主机数据库的连接是否存在问题。可设置您希望的正常错误阈值，这样每当错误数超过此阈值时都会进行通信错误调查。

将此元素与管理通知日志中记录的通信错误一起使用来确定问题。

相关参考:

- 第 446 页的『gw_comm_error_time – 通信错误时间』
- 第 372 页的『stmt_elapsed_time – 最新语句耗用时间』

gw_comm_error_time – 通信错误时间

元素标识 gw_comm_error_time

元素类型 时间戳记

表 788. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记

描述 DCS 应用程序尝试连接至主机数据库时或处理 SQL 语句时发生最新通信错误（SQL30081）的日期和时间。

用法 将此元素与通信错误及管理通知记录中记录的通信错误一起使用来确定问题。

相关参考:

- 第 445 页的『gw_comm_errors – 通信错误数』

blocking_cursor – 分块游标

元素标识 blocking_cursor

元素类型 信息

表 789. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 790. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-
语句	event_stmt	-

描述 此元素指示要执行的语句是否在使用分块游标。

用法 对查询的数据传输使用分块可以改进性能。用于查询的 SQL 会影响分块的使用并且可能需要一些修改。

事务处理器监视

事务处理器监视监视元素

在事务监视器或应用程序服务器（多层）环境中，应用程序用户不会直接发出 SQL 请求。它们会请求事务处理器监视器（例如，在 UNIX 或 Windows 服务器上运行的

CICS、TUXEDO 或 ENCINA）或应用程序服务器以执行业务交易。每个业务交易是一个应用程序部件，它将对数据库服务器发出 SQL 请求。因为中间服务器发出了 SQL 请求，所以数据库服务器没有导致执行 SQL 请求的原始客户机的相关信息。

事务处理器监视器（TP 监视器）事务或应用程序服务器代码的开发者可使用 sqleseti（设置客户机信息 API）来为数据库服务器提供有关原始客户机的信息。可在下列监视元素中找到此信息:
- tpmon_client_userid － TP 监视器客户机用户标识监视元素
- tpmon_client_wkstn － TP 监视器客户机工作站名监视元素
- tpmon_client_app － TP 监视器客户机应用程序名监视元素
- tpmon_acc_str － TP 监视器客户机记帐字符串监视元素.

tpmon_client_userid － TP 监视器客户机用户标识

元素标识 tpmon_client_userid

元素类型 信息

表 791. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

描述 如果使用 **sqleseti** API，则此项是由事务管理器生成的并且提供给服务器的客户机用户标识。

用法 在应用程序服务器或 TP 监视器环境中使用此元素来标识对其执行事务的最终用户。

相关参考:

- 第 447 页的『tpmon_client_wkstn － TP 监视器客户机工作站名称』
- 第 448 页的『tpmon_client_app － TP 监视器客户机应用程序名称』
- 第 448 页的『tpmon_acc_str － TP 监视器客户机记帐字符串』

tpmon_client_wkstn － TP 监视器客户机工作站名称

元素标识 tpmon_client_wkstn

元素类型 信息

表 792. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

描述 如果在此连接中发出了 **sqleseti** API，则此项标识客户机的系统或工作站（如 CICS EITERMID）。

用法 使用此元素并借助节点标识、终端标识或类似标识来标识用户的机器。

相关参考:

- 第 447 页的『tpmon_client_userid － TP 监视器客户机用户标识』

- 第 448 页的『tpmon_client_app – TP 监视器客户机应用程序名称』
- 第 448 页的『tpmon_acc_str – TP 监视器客户机记帐字符串』

tpmon_client_app – TP 监视器客户机应用程序名称

元素标识 tpmon_client_app

元素类型 信息

表 793. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

描述 如果在此连接中发出 **sqleseti** API，则此项标识执行事务时出现的服务器事务程序问题。

用法 此元素用于问题确定和记帐目的。

相关参考:

- 第 447 页的『tpmon_client_userid – TP 监视器客户机用户标识』
- 第 447 页的『tpmon_client_wkstn – TP 监视器客户机工作站名称』
- 第 448 页的『tpmon_acc_str – TP 监视器客户机记帐字符串』

tpmon_acc_str – TP 监视器客户机记帐字符串

元素标识 tpmon_acc_str

元素类型 信息

表 794. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

描述 如果在此连接中发出了 **sqleseti** API，则此项是为了用于记录和诊断而传递至目标数据库的数据。

用法 此元素用于问题确定和记帐目的。

相关参考:

- 第 447 页的『tpmon_client_userid – TP 监视器客户机用户标识』
- 第 447 页的『tpmon_client_wkstn – TP 监视器客户机工作站名称』
- 第 448 页的『tpmon_client_app – TP 监视器客户机应用程序名称』

联合数据库系统

联合数据库系统监视元素

联合系统是提供远程数据访问的多数据库服务器。它提供对各种数据源的客户机访问，这些数据源可驻留在不同平台上，包括 IBM® 和其他供应商平台、关系和非关系平台。它集成了对分布式数据的访问，并为多机种环境的用户提供了单一数据库映像。

下列元素列示了有关在 DB2 联合系统上运行的应用程序对数据源的总体访问的信息，以及有关在联合服务器实例上运行的给定应用程序对数据源的访问的信息。这些工具包括:

- datasource_name – 数据源名称监视元素
- disconnects – 断开连接数监视元素
- insert_sql_stmts – 插入数监视元素
- update_sql_stmts – 更新数监视元素
- delete_sql_stmts – 删除数监视元素
- create_nickname – 创建昵称监视元素
- passthrus – 传递监视元素
- stored_procs – 存储过程监视元素
- remote_locks – 远程锁定数监视元素
- sp_rows_selected – 存储过程返回的行数监视元素
- select_time – 查询响应时间监视元素
- insert_time – 插入响应时间监视元素
- update_time – 更新响应时间监视元素
- delete_time – 删除响应时间监视元素
- create_nickname_time – 创建昵称响应时间监视元素
- passthru_time – 传递时间监视元素
- stored_proc_time – 存储过程时间监视元素
- remote_lock_time – 远程锁定时间监视元素

datasource_name – 数据源名称

元素标识 datasource_name

元素类型 信息

表 795. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

描述 此元素包含其远程访问信息由联合服务器显示的数据源的名称。此元素对应于 SYSCAT.SERVERS 中的 'SERVER' 列。

用法 使用此元素来标识已收集并正在返回其访问信息的数据源。

disconnects － 断开连接次数

元素标识 断开连接次数

元素类型 计数器

表 796. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本

可将快照监视的计数器复位。

描述 此元素包含发生以下情况后联合服务器代表任何应用程序与此数据源断开连接的总次数的计数:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

用法 使用此元素来确定联合服务器代表任何应用程序与此数据源断开连接的总次数。此元素与 CONNECT 计数一起使用可提供一种机制，您可以通过这种机制来确定此联合服务器实例相信且当前已连接至数据源的应用程序数。

insert_sql_stmts － 插入数

元素标识 insert_sql_stmts

元素类型 计数器

表 797. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

描述 此元素包含发生以下情况后联合服务器代表任何应用程序对此数据源发出 INSERT 语句的总次数:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

用法 使用此元素来确定联合服务器或应用程序对此数据源执行的数据库活动的级别。

还可使用此元素并借助以下公式来确定联合服务器或应用程序对此数据源执行的写入活动的百分比:

```
write_activity =
   (INSERT 语句数 + UPDATE 语句数 + DELETE 语句数 ) /
   (SELECT 语句数 + INSERT 语句数 + UPDATE 语句数 +
   DELETE 语句数)
```

update_sql_stmts － 更新数

元素标识 update_sql_stmts

元素类型 计数器

表 798. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

描述 此元素包含发生以下情况后联合服务器代表任何应用程序对此数据源发出 UPDATE 语句的总次数:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

用法 使用此元素来确定联合服务器或应用程序对此数据源执行的数据库活动的级别。

还可使用此元素并借助以下公式来确定联合服务器或应用程序对此数据源执行的写入活动的百分比:

```
write_activity =
   (INSERT 语句数 + UPDATE 语句数 + DELETE 语句数 ) /
   (SELECT 语句数 + INSERT 语句数 + UPDATE 语句数 +
   DELETE 语句数)
```

delete_sql_stmts － 删除数

元素标识	delete_sql_stmts
元素类型	计数器

表 799. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

描述 此元素包含发生以下情况后联合服务器代表任何应用程序对此数据源发出 DELETE 语句的总次数:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

用法 使用此元素来确定联合服务器或应用程序对此数据源执行的数据库活动的级别。

还可使用此元素并借助以下公式来确定联合服务器或应用程序对此数据源执行的写入活动的百分比:

```
write_activity =
   (INSERT 语句数 + UPDATE 语句数 + DELETE 语句数 ) /
   (SELECT 语句数 + INSERT 语句数 + UPDATE 语句数 +
   DELETE 语句数)
```

create_nickname － 创建昵称数

元素标识	create_nickname
元素类型	计数器

表 800. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

描述 此元素包含发生以下情况后联合服务器代表任何应用程序对驻留在此数据源上的对象创建昵称的总次数:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

用法 使用此元素来确定此联合服务器实例或应用程序对此数据源执行的 CREATE NICKNAME 活动的级别。CREATE NICKNAME 处理将导致对数据源目录运行多个查询；因此，如果此元素的值很高，则应确定原因并在可能的情况下限制此活动的执行。

passthrus － 传递数

元素标识 passthrus

元素类型 计数器

表 801. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

描述 此元素包含发生以下情况后联合服务器代表任何应用程序直接传递至此数据源的 SQL 语句总数:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

用法 使用此元素来确定可由联合服务器在本地处理的 SQL 语句的百分比以及需要以通过方式处理的 SQL 语句的百分比。如果此值很高，则应确定原因和调查方法以便更好地利用本地支持。

stored_procs － 存储过程数

元素标识 stored_procs

元素类型 计数器

表 802. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

描述 此元素包含发生以下情况后联合服务器代表任何应用程序在此数据源上调用的存储过程总数:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

用法 使用此元素来确定联合数据库或应用程序在本地对联合数据库调用的存储过程数。

remote_locks － 远程锁定数

元素标识 remote_locks

元素类型 计数器

表 803. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

描述 此元素包含发生以下情况后联合服务器代表任何应用程序在此数据源上调用的远程锁定总数:

- 联合服务器实例启动
- 数据库监视器计数器的最后一次复位。

用法 使用此元素来确定在数据源上进行的远程锁定的数目。

sp_rows_selected － 存储过程返回的行数

元素标识 sp_rows_selected

元素类型 计数器

表 804. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

描述 此元素包含因为发生以下情况后对此应用程序执行存储过程操作而导致从数据源发送至联合服务器的行数:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

用法 此元素有若干用途。可使用它并借助以下公式来计算对于每个存储过程而言从数据源发送至联合服务器的平均行数:

```
        每个存储过程的行数
  = 返回的行数
  / 调用的存储过程数
```

还可以计算对于此应用程序而言行从数据源返回至联合服务器的平均时间:

平均时间 = 聚集存储过程响应时间 / 返回的行数

select_time － 查询响应时间

元素标识 select_time

元素类型 计数器

表 805. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

描述 此元素包含此数据源响应查询所花的聚集时间量（以毫秒计），这些查询来自发生以下情况以后在此联合服务器实例上运行的所有应用程序或单个应用程序:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

响应时间是以联合服务器请求数据源中的行的时间与该行可供联合服务器使用的时间之差量度的。

注: 因为查询分块，并非联合服务器检索行的所有尝试都能进行通信处理；所以获取下一行的请求可能要通过返回行块来得到处理。因此，聚集查询响应时间并不总是指示数据源上的处理，但它通常指示数据源或客户机上的处理。

用法 使用此元素来确定等待此数据源中的数据所花的实际时间。它在容量规划和容量调整 SYSCAT.SERVERS 中的 CPU 速度和通信速率时很有用。修改这些参数会影响优化器是否将请求发送至数据源。

insert_time － 插入响应时间

元素标识 insert_time

元素类型 计数器

表 806. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

描述 此元素包含此数据源响应 INSERT 语句所花的聚集时间量（以毫秒计），这些 INSERT 语句来自发生以下情况以后在此联合服务器实例上运行的所有应用程序或单个应用程序:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

响应时间是以联合服务器将 INSERT 语句提交给数据源的时间与数据源响应联合服务器以指示已处理 INSERT 的时间之差量度的。

用法 使用此元素来确定等待处理对此数据源执行 INSERT 语句所花的实际时间。此信息对于容量规划和容量调整非常有用。

update_time – 更新响应时间

元素标识 update_time

元素类型 计数器

表 807. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

描述 此元素包含此数据源响应 UPDATE 语句所花的聚集时间量（以毫秒计），这些 UPDATE 语句来自发生以下情况以后在此联合服务器实例上运行的所有应用程序或单个应用程序:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

响应时间是以联合服务器将 UPDATE 语句提交给数据源的时间与数据源响应联合服务器以指示已处理 UPDATE 的时间之差量度的。

用法 使用此元素来确定等待处理对此数据源执行 UPDATE 语句所花的实际时间。此信息对于容量规划和容量调整非常有用。

delete_time – 删除响应时间

元素标识 delete_time

元素类型 计数器

表 808. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

描述 此元素包含此数据源响应 DELETE 语句所花的聚集时间量（以毫秒计），这些 DELETE 语句来自发生以下情况以后在此联合服务器实例上运行的所有应用程序或单个应用程序:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

响应时间是以联合服务器将 DELETE 语句提交给数据源的时间与数据源响应联合服务器以指示已处理 DELETE 的时间之差量度的。

用法 使用此元素来确定等待处理对此数据源执行 DELETE 语句所花的实际时间。此信息对于容量规划和容量调整非常有用。

create_nickname_time － 创建昵称响应时间

元素标识 create_nickname_time

元素类型 计数器

表 809. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

描述 此元素包含此数据源处理 CREATE NICKNAME 语句所花的聚集时间量（以毫秒计），这些 CREATE NICKNAME 语句来自发生以下情况以后在此联合服务器实例上运行的所有应用程序或单个应用程序:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

响应时间是以联合服务器开始从数据源接收信息以处理 CREATE NICKNAME 语句的时间与检索数据源中的所有必需数据所花时间之差量度的。

用法 使用此元素来确定用于为此数据源创建昵称所花的实际时间。

passthru_time － 传递时间

元素标识 passthru_time

元素类型 计数器

表 810. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

描述 此元素包含此数据源响应 PASSTHRU 语句所花的聚集时间量（以毫秒计），这些 PASSTHRU 语句来自发生以下情况以后在此联合服务器实例上运行的所有应用程序或单个应用程序:

- 联合服务器实例启动，或者
- 数据库监视器计数器的最后一次复位。

响应时间是以联合服务器将 PASSTHRU 语句提交给数据源的时间与数据源响应以指示已处理 PASSTHRU 语句的时间之差量度的。

用法 使用此元素来确定在此数据源上以通过方式处理语句所花的实际时间。

stored_proc_time － 存储过程时间

元素标识 stored_proc_time

元素类型 计数器

表 811. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

描述 此元素包含此数据源响应存储过程语句所花的聚集时间量（以毫秒计）， 这些存储过程语句来自发生以下情况以后在此联合服务器实例上运行的所有应用程序或单个应用程序:

- 联合服务器实例启动
- 数据库监视器计数器的最后一次复位。

响应时间是以联合服务器将存储过程提交给数据源的时间与数据源响应以指示已处理存储过程的时间之差量度的。

用法 使用此元素来确定在此数据源上处理存储过程所花的实际时间。

remote_lock_time – 远程锁定时间

元素标识 remote_lock_time

元素类型 计数器

表 812. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

描述 此元素包含此数据源在远程锁定上耗用的聚集时间量（以毫秒计）， 该远程锁定来自发生以下情况以后在此联合服务器实例上运行的所有应用程序或单个应用程序:

- 联合服务器实例启动， 或者
- 数据库监视器计数器的最后一次复位。

响应时间是以联合服务器将远程锁定提交给数据源的时间与联合服务器在数据源上释放远程锁定的时间之差量度的。

用法 使用此元素来确定此数据源对远程锁定所花的实际时间。

第 7 章 监视器接口

数据库系统监视器接口

监视任务	API
捕获快照	db2GetSnapshot API - Get a snapshot of the database manager operational statusdb2GetSnapshot API - Get a snapshot of the database manager operational status
转换自描述数据流	db2ConvMonStream API - Convert the monitor stream to the pre-version 6 formatdb2ConvMonStream API - Convert the monitor stream to the pre-version 6 format
显示数据库系统监视开关	db2MonitorSwitches API - Get or update the monitor switch settings
估计快照大小	db2GetSnapshotSize API - Estimate the output buffer size required for the db2GetSnapshot APIdb2GetSnapshotSize API - Estimate the output buffer size required for the db2GetSnapshot API
获取／更新监视开关	db2MonitorSwitches API - Get or update the monitor switch settingsdb2MonitorSwitches API - Get or update the monitor switch settings
复位监视器计数器	db2ResetMonitor API - Reset the database system monitor datadb2ResetMonitor API - Reset the database system monitor data
更新数据库系统监视开关	db2MonitorSwitches API - Get or update the monitor switch settings

监视任务	CLP 命令
使用 GUI 工具分析事件监视器输出	db2eva - Event analyzer commanddb2eva - Event analyzer command
捕获快照	GET SNAPSHOT commandGET SNAPSHOT command
显示数据库管理器系统监视开关	GET DATABASE MANAGER MONITOR SWITCHES commandGET DATABASE MANAGER MONITOR SWITCHES command
显示监视应用程序的监视开关	GET MONITOR SWITCHES commandGET MONITOR SWITCHES command
格式化事件监视器跟踪	db2evmon - Event monitor productivity tool commanddb2evmon - Event monitor productivity tool command
对写至表 CREATE EVENT MONITOR 语句生成样本 SQL	db2evtbl
列示活动数据库	LIST ACTIVE DATABASES commandLIST ACTIVE DATABASES command
列示连接至数据库的应用程序	LIST APPLICATIONS commandLIST APPLICATIONS command
列示 DCS 应用程序	LIST DCS APPLICATIONS commandLIST DCS APPLICATIONS command
复位监视器计数器	RESET MONITOR commandRESET MONITOR command
更新数据库系统监视开关	UPDATE MONITOR SWITCHES commandUPDATE MONITOR SWITCHES command

监视任务	SQL 语句
激活事件监视器	SET EVENT MONITOR STATE statementSET EVENT MONITOR STATE statement
创建事件监视器	CREATE EVENT MONITOR statementCREATE EVENT MONITOR statement
释放事件监视器	SET EVENT MONITOR STATE statementSET EVENT MONITOR STATE statement
除去事件监视器	DROP statementDROP statement
写入事件监视器值	FLUSH EVENT MONITOR statementFLUSH EVENT MONITOR statement

监视任务	SQL 函数
确定事件监视器的状态	EVENT_MON_STATE scalar functionEVENT_MON_STATE scalar function
获取数据库管理器级别快照	SNAPDBM administrative view and SNAP_GET_DBM table function – Retrieve the dbm logical grouping snapshot informationSNAPDBM administrative view and SNAP_GET_DBM table function – Retrieve the dbm logical grouping snapshot information
在数据库管理器级别获取当前监视开关设置	SNAPSWITCHES administrative view and SNAP_GET_SWITCHES table function – Retrieve database snapshot switch state informationSNAPSWITCHES administrative view and SNAP_GET_SWITCHES table function – Retrieve database snapshot switch state information
获取快速通信管理器快照	SNAPFCM administrative view and SNAP_GET_FCM table function – Retrieve the fcm logical data group snapshot informationSNAPFCM administrative view and SNAP_GET_FCM table function – Retrieve the fcm logical data group snapshot information
获取给定分区的快速通信管理器快照	SNAPFCM_PART administrative view and SNAP_GET_FCM_PART table function – Retrieve the fcm_node logical data group snapshot informationSNAPFCM_PART administrative view and SNAP_GET_FCM_PART table function – Retrieve the fcm_node logical data group snapshot information
获取数据库级别快照	SNAPDB administrative view and SNAP_GET_DB_V91 table function – Retrieve snapshot information from the dbase logical groupSNAPDB administrative view and SNAP_GET_DB_V91 table function – Retrieve snapshot information from the dbase logical group
获取应用程序级别快照	SNAPAPPL administrative view and SNAP_GET_APPL table function – Retrieve appl logical data group snapshot informationSNAPAPPL administrative view and SNAP_GET_APPL table function – Retrieve appl logical data group snapshot information
获取应用程序级别快照	SNAPAPPL_INFO administrative view and SNAP_GET_APPL_INFO table function – Retrieve appl_info logical data group snapshot informationSNAPAPPL_INFO administrative view and SNAP_GET_APPL_INFO table function – Retrieve appl_info logical data group snapshot information

监视任务	SQL 函数
获取锁定等待信息的应用程序级别快照	SNAPLOCKWAIT administrative view and SNAP_GET_LOCKWAIT table function – Retrieve lockwait logical data group snapshot informationSNAPLOCKWAIT administrative view and SNAP_GET_LOCKWAIT table function – Retrieve lockwait logical data group snapshot information
获取语句信息的应用程序级别快照	SNAPSTMT administrative view and SNAP_GET_STMT table function – Retrieve statement snapshot informationSNAPSTMT administrative view and SNAP_GET_STMT table function – Retrieve statement snapshot information
获取代理程序信息的应用程序级别快照	SNAPAGENT administrative view and SNAP_GET_AGENT table function – Retrieve agent logical data group application snapshot informationSNAPAGENT administrative view and SNAP_GET_AGENT table function – Retrieve agent logical data group application snapshot information
获取子节信息的应用程序级别快照	SNAPSUBSECTION administrative view and SNAP_GET_SUBSECTION table function – Retrieve subsection logical monitor group snapshot informationSNAPSUBSECTION administrative view and SNAP_GET_SUBSECTION table function – Retrieve subsection logical monitor group snapshot information
获取缓冲池级别快照	SNAPBP administrative view and SNAP_GET_BP table function – Retrieve bufferpool logical group snapshot informationSNAPBP administrative view and SNAP_GET_BP table function – Retrieve bufferpool logical group snapshot information
获取表空间级别快照	SNAPTBSP administrative view and SNAP_GET_TBSP_V91 table function – Retrieve tablespace logical data group snapshot informationSNAPTBSP administrative view and SNAP_GET_TBSP_V91 table function – Retrieve tablespace logical data group snapshot information
获取配置信息的表空间级别快照	SNAPTBSP_PART administrative view and SNAP_GET_TBSP_PART_V91 table function – Retrieve tablespace_nodeinfo logical data group snapshot informationSNAPTBSP_PART administrative view and SNAP_GET_TBSP_PART_V91 table function – Retrieve tablespace_nodeinfo logical data group snapshot information
获取容器信息的表空间级别快照	SNAPCONTAINER administrative view and SNAP_GET_CONTAINER_V91 table function – Retrieve tablespace_container logical data group snapshot informationSNAPCONTAINER administrative view and SNAP_GET_CONTAINER_V91 table function – Retrieve tablespace_container logical data group snapshot information
获取停顿者信息的表空间级别快照	SNAPTBSP_QUIESCER administrative view and SNAP_GET_TBSP_QUIESCER table function – Retrieve quiescer table space snapshot informationSNAPTBSP_QUIESCER administrative view and SNAP_GET_TBSP_QUIESCER table function – Retrieve quiescer table space snapshot information

监视任务	SQL 函数
获取表空间映射范围的表空间级别快照	SNAPTBSP_RANGE administrative view and SNAP_GET_TBSP_RANGE table function – Retrieve range snapshot informationSNAPTBSP_RANGE administrative view and SNAP_GET_TBSP_RANGE table function – Retrieve range snapshot information
获取表级别快照	SNAPTAB administrative view and SNAP_GET_TAB_V91 table function – Retrieve table logical data group snapshot informationSNAPTAB administrative view and SNAP_GET_TAB_V91 table function – Retrieve table logical data group snapshot information
获取锁定级别快照	SNAPLOCK administrative view and SNAP_GET_LOCK table function – Retrieve lock logical data group snapshot informationSNAPLOCK administrative view and SNAP_GET_LOCK table function – Retrieve lock logical data group snapshot information
获取 SQL 语句高速缓存的快照	SNAPDYN_SQL administrative view and SNAP_GET_DYN_SQL_V91 table function – Retrieve dynsql logical group snapshot informationSNAPDYN_SQL administrative view and SNAP_GET_DYN_SQL_V91 table function – Retrieve dynsql logical group snapshot information

相关参考:

- 『CREATE EVENT MONITOR statement』 (*SQL Reference, Volume 2*)
- 『db2ConvMonStream API - Convert the monitor stream to the pre-version 6 format』 (*Administrative API Reference*)
- 『db2eva - Event analyzer command』 (*Command Reference*)
- 『db2evmon - Event monitor productivity tool command』 (*Command Reference*)
- 『db2GetSnapshot API - Get a snapshot of the database manager operational status』 (*Administrative API Reference*)
- 『db2GetSnapshotSize API - Estimate the output buffer size required for the db2GetSnapshot API』 (*Administrative API Reference*)
- 『db2MonitorSwitches API - Get or update the monitor switch settings』 (*Administrative API Reference*)
- 『db2ResetMonitor API - Reset the database system monitor data』 (*Administrative API Reference*)
- 『EVENT_MON_STATE scalar function』 (*SQL Reference, Volume 1*)
- 『FLUSH EVENT MONITOR statement』 (*SQL Reference, Volume 2*)
- 『GET DATABASE MANAGER MONITOR SWITCHES command』 (*Command Reference*)
- 『GET MONITOR SWITCHES command』 (*Command Reference*)
- 『GET SNAPSHOT command』 (*Command Reference*)
- 『LIST ACTIVE DATABASES command』 (*Command Reference*)
- 『LIST APPLICATIONS command』 (*Command Reference*)
- 『LIST DCS APPLICATIONS command』 (*Command Reference*)
- 『RESET MONITOR command』 (*Command Reference*)

- 『SET EVENT MONITOR STATE statement』（*SQL Reference, Volume 2*）
- 『SYSCAT.EVENTMONITORS catalog view』（*SQL Reference, Volume 1*）
- 『SYSCAT.EVENTS catalog view』（*SQL Reference, Volume 1*）

活动监控器概述

使用“活动监控器”来监视应用程序性能和并行性、资源消耗以及数据库或数据库分区的 SQL 语句使用情况。活动监控器提供一组基于特定监视器数据子集的预定义报告。这些报告允许您将重点放在监视应用程序性能、应用程序并行性、资源消耗和 SQL 语句使用情况上。“活动监控器”还为大多数报告提供了建议。这些建议可以帮助您诊断发生数据库性能问题的原因，并且调整查询以便最佳地使用数据库资源。

第 464 页的图 5描述使用活动监控器解决问题的过程。

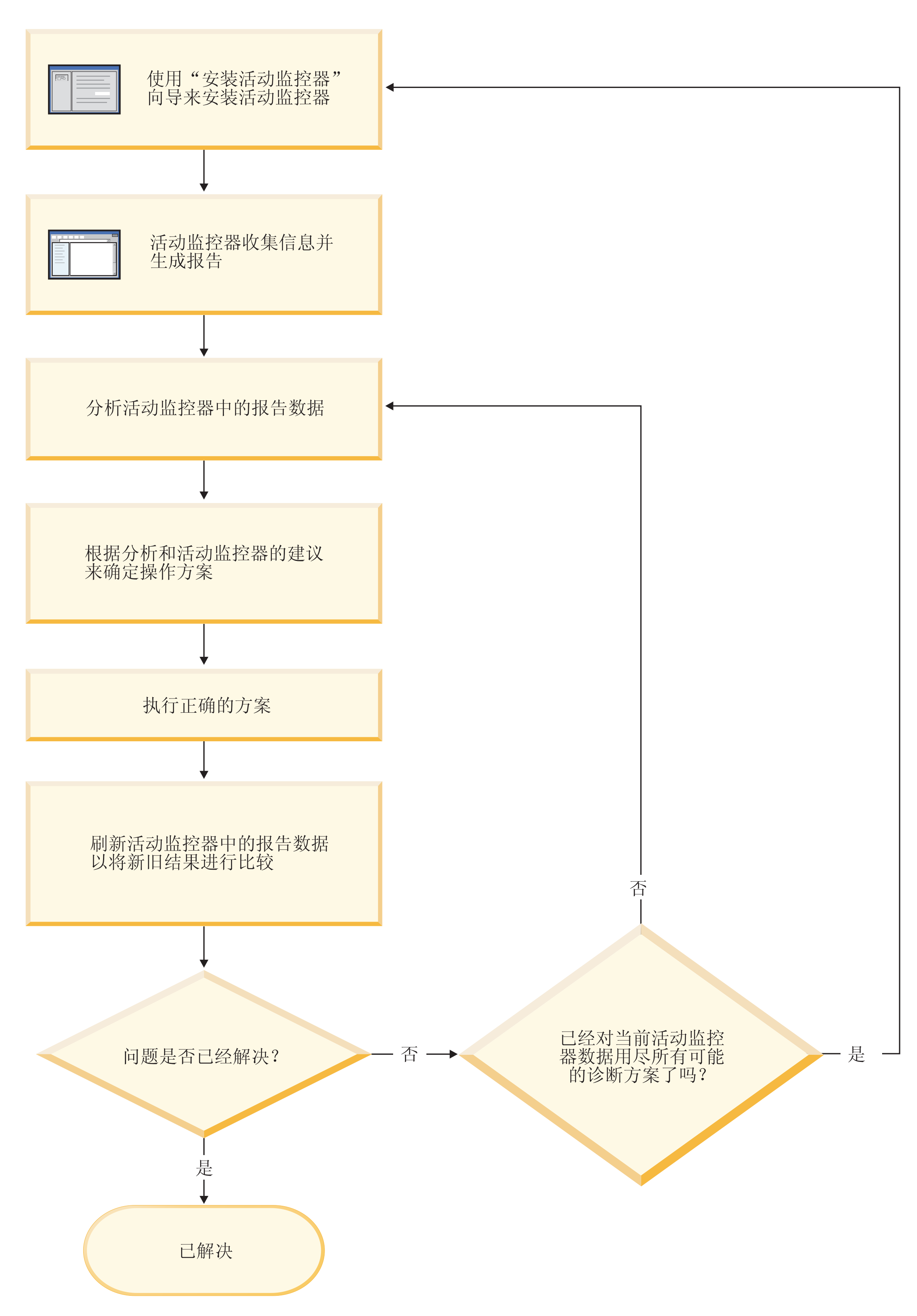


图 5. 活动监控器概述

先决条件:

当您想要使用活动监控器监视数据库活动或解决特定问题时，第一步通常是调用“设置活动监控器”向导。要了解更多信息，请参阅设置活动监控器。

一次可打开多个“活动监控器”窗口。

活动监控器中的任务:

表 813. 可在活动监控器中执行的任务

活动监控器中的任务	任务的各个方面	调用
事务	查看在所选应用程序上运行的事务。	在**报告数据**窗格中选择一个或多个应用程序。右键单击并选择**显示最新事务**。“应用程序事务”窗口将打开。
语句	查看在所选应用程序上运行的 SQL 语句。	在**报告数据**窗格中选择一个或多个应用程序。右键单击并选择**显示最新语句**。“应用程序语句”窗口将打开。
	查看在所选应用程序上运行的 SQL 语句的文本。	从“应用程序语句”窗口中，右键单击**报告数据**窗格中的语句。选择**显示语句文本**。
应用程序锁定链	查看当前影响所选应用程序的锁定和锁定等待情况。	在**报告数据**窗格中选择应用程序。右键单击并选择**显示锁定链**。“应用程序锁定链”窗口将打开。
	查看有关所选应用程序的信息，您正在查看该应用程序的锁定信息。	从“应用程序锁定链”窗口右键单击应用程序，然后选择**关于**。
	查看有关数据库中所选应用程序挂起的锁定和等待的锁定的信息。	从“应用程序锁定链”窗口右键单击应用程序，然后选择**显示锁定详细信息**。
查看报告数据和建议	查看可帮助您解释报告数据的信息。	在“活动监控器”窗口、“应用程序语句”窗口或“应用程序事务”窗口中，使用**报告**箭头键来选择该报告并单击**报告详细信息**按钮。查看**详细信息**页。
	查看活动监控器提供的建议	在“活动监控器”窗口、“应用程序语句”窗口或“应用程序事务”窗口中，使用**报告**箭头键来选择该报告并单击**报告详细信息**按钮。查看**建议**页。

活动监控器界面:

“活动监控器”界面中有一些元素，可使用这些元素来帮助您组织和解释收集到的监视器数据。

菜单栏

Activity Monitor - Tuning dynamic SQL statement cache
Activity Monitor Selected Tools Help

使用菜单栏在“活动监控器”中处理对象，打开其他管理中心工具和访问联机帮助。

活动监控器工具栏



使用工具栏图标来打开 DB2 工具和查看 DB2 信息。

报告数据窗格

Report data

Application Handle (agent ID)	Application Name	Authorization ID	Application ID	Total CPU Time	User CPU Time
18	acmerpt.exe	EDWARDL	*LOCAL.DB2.00...	180259	10014
20	db2cc.exe	DB2ADMIN	*LOCAL.DB2.00...	30042	10014
22	acmefin.exe	FREDS	*LOCAL.DB2.00...	20028	20028
21	db2evm.exe	DB2ADMIN	*LOCAL.DB2.00...	20028	10014
27	acmeacct.exe	ALICET	*LOCAL.DB2.00...	10015	10015

使用**报告数据**窗格来显示和处理“活动监控器”中提供的报告数据。**报告数据**窗格显示构成**报告**字段中选择的报告内容的各项。

报告数据窗格还会提供对其他“活动监控器”窗口的访问。在“活动监控器”中，可从要监视的应用程序向下钻取至各个事务，或向下钻取至这些应用程序要运行的各个 SQL 语句。

报告数据窗格工具栏



使用**报告数据**窗格下面的工具栏来调整**报告数据**窗格中的对象和信息的视图，以满足您的需要。

相关任务:

- 第 466 页的『设置活动监控器』

设置活动监控器

要监视应用程序性能和并行性、资源消耗以及数据库或数据库分区的 SQL 语句使用情况，可设置活动监控器。活动监控器提供一组基于特定监视器数据子集的预定义报告。它还提供一些建议，以帮助您诊断数据库性能问题的原因，以及调整查询以优化数据库资源的使用情况。

先决条件:

要使用活动监控器:

- 您的服务器上必须有 DB2 UDB 版本 8.2 或更高版本
- 您必须有 DBADM 权限

过程:

可通过控制中心中的“设置活动监控器”向导来设置活动监控器。要从命令行打开“设置活动监控器”向导，请输入以下命令：**db2am**。“设置活动监控器”向导将打开。

要从控制中心打开“设置活动监控器”向导：

1. 展开对象树直到找到想要对其设置活动监控器的实例或数据库。
2. 右键单击该对象，然后从弹出菜单中选择“设置活动监控器”。“设置活动监控器”向导将打开。

在“控制中心”中通过上下文帮助工具提供详细信息。

相关概念:

- 第 463 页的『活动监控器概述』

内存可视化器概述

使用“内存可视化器”来监视实例及其所有数据库的内存相关性能。

打开“内存可视化器”并在分层树中选择一个或多个内存组件，以在“内存可视化器”窗口中显示对应分配给该组件的内存量及当前内存使用情况的值。“内存可视化器”窗口显示两个数据视图：树形视图和历史视图。一系列的列显示警报和警告的上限和下限的百分比阈值。这些列还将显示实时内存利用率。

注: “内存可视化器”可用来为版本 8.1 和更新版本的实例提供内存性能数据。

要从 Windows **开始**菜单打开“内存可视化器”：单击**程序** → **IBM DB2** → **监视** → **工具** → **内存可视化器**。“内存可视化器”实例选择窗口将打开。从**实例名**字段选择实例并单击**确定**。

要从控制中心打开“内存可视化器”：展开对象树直至找到想要查看的实例。右键单击该实例并从弹出菜单中选择**查看内存使用情况**。

内存可视化器中的任务:

以下列表对可在“内存可视化器”中执行的一些关键任务进行了分类:

查看或隐藏有关 DB2 实例及其数据库的所选组件的内存利用率的各列中的数据。
查看内存性能数据图。
通过更新配置参数来更改各个内存组件的设置。
将文件中的性能数据装入到“内存可视化器”窗口中。
保存内存性能数据。

有关执行上述任务的详细信息，请参阅使用内存可视化器。

内存可视化器界面:

“内存可视化器”界面具有下列元素，可使用它们来帮助您监视实例及其所有数据库的内存相关性能。

内存可视化器窗口

“内存可视化器”窗口中的各列显示对应内存组件的性能的值。显示下列信息:

绘制图注
显示在“内存使用情况绘图”中的所选内存组件或配置参数。按一定时间间隔出现在已绘制的图中的特定形状标识每个组件或参数。

利用率 分配给数据库对象或被数据库对象利用的内存的大小。包括显示利用率和已配置分配的图形柱。该柱的长度是固定的，填充部分以百分比的形式指示利用率。

参数值 配置参数的当前值。

警报上限（%）阈值
生成警报上限的阈值。缺省值为 98%。

警告上限（%）阈值
生成警告上限的阈值。缺省值为 90%。

警报下限（%）阈值
生成警报下限的阈值。缺省值为 2%。

警告下限（%）阈值
生成警告下限的阈值。缺省值为 10%。

图形使用柱

“内存可视化器”窗口中的图形使用柱是内存利用率的可视表示。这些柱可帮助您确定所选内存组件要使用的内存量及该使用量对系统造成的潜在影响。“内存可视化器”还会显示对应于使用情况的百分比值。这两个指示器可帮助您确定是否需要更改组件的配置参数设置或采用另一适当操作。

内存组件

“数据库管理器”在系统上使用以下不同类型的内存：数据库管理器共享内存、数据库全局内存、应用程序全局内存、代理程序／应用程序共享内存和代理程序专用内存。这些内存类型是“内存可视化器”在展开的分层树结构中使用的高级内存组件。

每个高级内存组件下面是用来确定分配或释放内存的方式的其他组件。例如，当数据库管理器启动时，将激活数据库，而应用程序将连接至数据库，或者指定代理程序以便为应用程序工作时，将分配和释放内存。“内存可视化器”使用这些叶级别内存组件来显示在 DB2 实例中分配和使用内存的方式。有关 DB2 使用内存的方式的更多信息，请参阅《管理指南》。

分层树结构

“内存可视化器”使用分层树结构来帮助您显示和浏览 DB2 中的内存组件。分层树允许您展开和查看各个内存组件的列、图形显示和图的信息。

树形视图由四个主要内存项类型组成：

DB2实例
当前在系统上运行的实例

数据库 在实例上定义的数据库

高级内存组件
叶级别内存组件的逻辑分组。这些组包括：数据库管理器共享内存、数据库全局内存、代理程序专用内存和代理程序／应用程序共享内存。

叶级别内存组件

“内存可视化器”窗口中显示的内存组件，如缓冲池、排序堆、数据库堆和锁定列表。

树形视图中的图标表示每个内存树项:

- 实例:
- 数据库:
- 高级内存分组:
- 叶级别内存组件:

如果树项的内存利用率超过阈值，则彩色指示器会覆盖图标。黄色指示警告条件。红色指示警报条件。

历史视图显示树形视图中选择的内存组件的数据。该数据包括分配的和利用的内存的值，已绘制的图以及运行“内存可视化器”时对配置参数所作的更改。将在“内存可视化器”中的特定时间段内保存该数据。可将内存性能数据保存至“内存可视化器”数据文件，以便跟踪、与其他数据进行比较或进行故障诊断。

内存使用情况绘图

内存使用情况绘图显示所选内存组件在“内存使用情况绘图”中的已绘制的数据。图中的每个组件由一种特定颜色标识，它还会显示在“内存可视化器”窗口的**绘制图注**列中。该图还会显示对配置参数设置的更改。除了请求更改的时间外，配置参数的初始值和新值设置也将显示在图中。它们将成为可在评估内存性能时使用的历史视图的一部分。

有关更多信息，请参阅使用内存可视化器。

用键盘访问定制控件:

可使用键盘来访问用户界面上的控件。

要了解更多信息，请参阅键盘快捷键和加速键（所有中心）。

相关任务:

- 第 469 页的『使用内存可视化器』

使用内存可视化器

“内存可视化器”帮助数据库管理员监视实例及其所有数据库的内存相关性能。可以查看以分层树结构显示的内存组件的内存利用率的即时可视显示。

还可使用“内存可视化器”来对性能问题进行故障诊断。可更改内存组件的配置参数设置并评估更改的效果。因为内存是按需分配的，所以配置参数会影响 DB2 中的内存使用情况。如果将配置参数的值设置在可接受范围以外，则会显示错误消息。更改配置参数会直接影响“内存可视化器”，并且新值将集成到下一次刷新周期中。

要查看内存性能和使用情况绘图及更新“内存可视化器”中的配置参数，必须具有 SYSADM 权限。

要使用内存可视化器查看内存性能：：

1. 展开实例对象树直到分层树中显示数据库及其关联内存组件。内存池的值显示在“内存可视化器”窗口中。
2. 要显示内存组件的已绘制的图，请使用下列其中一个方法：
   a. 在分层树中选择一个组件，然后在“内存可视化器”窗口中单击**显示图形**复选框。
   b. 右键单击所选内存组件以显示弹出菜单并选择**显示图形**。
   c. 在分层树中选择一个组件，然后从“已选择”菜单的工具栏上选择**显示图形**选项。每个内存组件的已绘制的图数据出现在“内存使用情况绘图”中。
   d. 要查看另一内存组件中的数据，从分层树中选择该组件，然后单击**显示图形**复选框。该内存组件的已绘制的数据与其他组件一起出现在“内存使用情况绘图”中。

   该图显示在一段时间内为内存组件收集的数据。每个组件用它们在“内存可视化器”窗口的**绘制图注**字段中显示的颜色和形状来表示。形状按一定时间间隔重复。标签用来标识图形绘制中的组件。

   捕获性能数据的时间显示在图的下面。可以更改该图的时间间隔。

   注: 将新内存组件添加至绘图时，它不会替换先前添加的内存组件。

   水平和垂直滚动条允许您从不同角度查看已绘制的数据。
   - 使用位于图底部的水平滚动条，以查看所选时间段内的内存组件历史数据。按住滑块并沿着图的底边拖动滑块。
   - 使用位于图右边的垂直滚动条，以查看所选组件的内存利用率。按住滑块并拖动滑块以更改视图。

   内存利用率达到新的高度时，垂直滚动条的最大值将更新以反映新值。可将垂直滚动条的最小值设置为 0 以外的值以查看另一范围的池利用率值。
3. 可将“内存可视化器”数据文件中的数据装入到新的“内存可视化器”窗口中。此数据可用来针对历史数据比较实例及其所有数据库的性能。要从“内存可视化器”数据文件装入数据，从“内存可视化器”菜单中选择**打开**，然后从“打开对话框”中选择带有扩展名 *.mdf 的数据文件。
4. 使用**时间单位**字段在“内存使用情况绘图”窗口中更改时间间隔。图数据的缺省时间间隔以分钟计。可选择以分钟、小时或天为单位的时间间隔。选择后，新的时间间隔将显示在该图的水平范围中并更改水平滚动条的增量移动幅度。
5. 要从“内存使用情况绘图”中除去内存组件的已绘制的图，在分层树中选择一个组件并清除“内存可视化器”窗口中的**显示图形**复选框，或者右键单击所选内存组件以显示弹出菜单并取消选择**显示图形**。将从“内存使用情况绘图”窗口中除去该组件的已绘制的数据。表示组件的彩色形状不再显示在“内存可视化器”窗口的**绘制图注**字段中。
6. 为帮助您跟踪和创建内存性能的历史记录，可在“内存可视化器”运行时保存内存性能数据，包括已绘制的图。要保存内存性能数据，从“内存可视化器”菜单中选择**保存**或**另存为**，然后选择文件位置及带有扩展名 `.mdf` 的文件名。

要更改内存组件的配置参数设置：:

1. 展开想要的内存池，这样就会看到其配置参数列示在分层树中。
2. 单击组件以选择该组件，然后单击**参数值**列中的数字。文本框将显示该组件的当前值。在文本框中输入新数字并按 **Enter** 键。新值显示在**参数值**列中的原始值旁边，直到下一次刷新周期可能更新配置参数为止。还可右键单击**参数值**列中对应所选组件的值以显示弹出菜单。单击列的外部以完成更改。

内存组件的新值显示在**参数值**列中的原始值旁边。如果选择查看内存性能图，则会在视图中看到新值。虽然此更改将立即在“内存可视化器”中生效，但在 DB2 中更新对配置参数的更改时还是会有延迟。

可使用弹出菜单中的**复位为缺省值**选项复位配置参数的值。

相关概念:

- 第 467 页的『内存可视化器概述』

不确定事务管理器概述

使用“不确定事务管理器”窗口来处理不确定事务。该窗口列示所选数据库和一个或多个所选分区的所有不确定事务。

不确定事务是处于不确定状态的全局事务。当资源所有者（如数据库管理员）不能等待事务管理器执行再同步操作时， DB2 将提供数据库管理员可对不确定事务执行的试探操作。例如，如果通信线路中断，并且不确定事务正在安排需要的资源（如针对表和索引的锁定、日志空间和事务使用的存储器），则可能会出现这种情况。

虽然由事务管理器启动再同步操作比较好，但有时可能必须对不确定事务执行试探操作。在这些情况下，使用试探操作时应特别谨慎，不到万不得已不要使用，并且应遵循这些准则。

- 事务标识的 *gtrid* 部分是参与全局事务的其他资源管理器（RM）中的全局事务标识。
- 使用您掌握的有关该应用程序和操作环境的知识，来确定其他参与的资源管理器，
- 如果事务管理器为 CICS，并且唯一资源管理器为 CICS 资源，则执行试探回滚。
- 如果事务管理器不是 CICS，则使用它来确定与不确定事务具有相同 *gtrid* 的事务的状态。
- 如果至少有一个资源管理器已经落实或回滚，则执行试探落实或回滚。
- 如果所有事务处于已编译状态，则执行试探回滚。
- 如果至少有一个资源管理器不可用，则执行试探回滚。

要在 Intel 平台上打开不确定事务管理器，从**开始**菜单单击**开始 → 程序 →IBM DB2 → 监视工具 → 不确定事务管理器**。

要在 UNIX 或 Intel 上使用命令行打开不确定事务管理器，请运行以下命令:

```
db2indbt
```

不确定事务管理器中的任务:

可对不确定事务执行试探操作:

- 遗忘

它允许资源管理器通过除去日志记录并释放日志页来擦除试探性完成的事务的痕迹。试探性完成的事务就是试探性落实或回滚的事务。对于所选数据库及一个或多个所选分区，可对试探性落实或回滚的事务使用遗忘操作。要遗忘不确定事务，选择数据库和分区，然后右键单击状态为**已落实**或**已回滚**的事务，然后从弹出菜单中选择**遗忘**。将显示确认消息。

- 落实

  这会落实准备好落实的不确定事务。如果操作成功，事务的状态将变为试探性落实。要落实不确定事务，选择数据库和分区，然后右键单击状态为**不确定**或**落实确认已丢失**的事务，然后从弹出菜单中选择**落实**。将显示确认消息。

- 回滚

  这会回滚已准备好的不确定事务。如果操作成功，事务的状态将变为试探性回滚。要回滚不确定事务，选择数据库和分区，然后右键单击状态为**不确定**或**已结束**的事务，然后从弹出菜单中选择**回滚**。将显示确认消息。

要对不确定事务执行这些操作，必须具有 SYSADM 或 DBADM 权限。

不确定事务管理器界面:

“不确定事务管理器”窗口中的各列将提供指定视图，可使用这些视图来以不同方式组织和显示不确定事务。

不确定事务管理器窗口中的列:

状态 事务的不确定状态包括：已落实（c）、已结束（e）、不确定（i）、落实确认已丢失（m）和已回滚（r）：

已落实 处于此状态的事务已试探性落实。

已结束 处于此状态的事务可能已超时。

不确定 处于此状态的事务等待落实或回滚。

落实确认已丢失
事务管理器在落实事务之前等待接收确认。

已回滚 处于此状态的事务已试探性回滚。

时间戳记
事务进入已编译（不确定）状态时服务器上的时间戳记。该时间是客户机的本地时间。

事务标识
事务管理器为唯一标识全局事务而指定的 XA 标识。

应用程序标识
数据库管理器为此事务指定的应用程序标识。

授权标识
运行该事务的用户的标识。

序号 数据库管理器指定为应用程序标识扩展的序号。

分区 不确定事务所在的分区。

发起方 指示是 XA 还是 DB2 在分区数据库环境中发起事务。

日志满载
指示此事务是否导致日志满载情况。

类型 显示数据库在每个不确定事务中的角色的类型信息。
- **TM** 指示不确定事务将该数据库用作事务管理器数据库。
- **RM** 指示不确定事务将该数据库用作资源管理器。这表示它是参与事务的数据库之一，但不是事务管理器数据库。

用键盘访问定制控件:

可使用键盘来访问用户界面上的控件。

要了解更多信息，请参阅键盘快捷键和加速键（所有中心）。

相关参考:

- 『LIST DRDA INDOUBT TRANSACTIONS command』（*Command Reference*）
- 『LIST INDOUBT TRANSACTIONS command』（*Command Reference*）

第 3 部分 运行状况监视器指南

第 8 章 运行状况监视器简介

运行状况监视器简介

运行状况监视器是一个服务器端工具，它增添的“按异常情况进行管理”功能是通过不断监视实例和活动数据库的运行状况实现的。运行状况监视器还可以向数据库管理员（DBA）发出有关潜在系统运行状况问题的警报。运行状况监视器以前摄方式检测可能导致硬件故障或不可接受的系统系统性能或功能的问题。运行状况监视器的前摄特征使用户能够在发现的问题影响系统性能之前解决该问题。

运行状况监视器使用运行状况指示器来检查系统状态，以确定是否应该发出警报。可对应警报执行先前配置的操作。运行状况监视器还会将警报记录在管理通知日志中，并通过电子邮件或寻呼机发送通知。这一“按异常情况进行管理”模型对潜在系统运行状况问题生成警报而不需要活动监视，从而让 DBA 从事更有价值的工作。

运行状况监视器定期收集有关系统运行状况的信息，这对整体性能的影响非常小。它不会打开任何快照监视开关来收集信息。

相关概念:

- 第 479 页的『运行状况指示器处理周期』
- 第 485 页的『运行状况监视器』

相关任务:

- 第 481 页的『启用运行状况报警通知』

相关参考:

- 第 477 页的『运行状况指示器』

运行状况指示器

运行状况监视器使用运行状况指示器来评估数据库管理器性能或数据库性能特定方面的运行状况。运行状况指示器测量特定种类数据库对象（例如表空间）某些方面的运行状况。对度量应用条件以确定运行状况。应用的条件视运行状况指示器类型而定。根据生成警报的条件来确定运行状况是否不良。

运行状况监视器返回三种类型的运行状况指示器:

- **基于阈值**的指示器是对象行为的（连续值范围）统计信息度量。警告阈值和警报阈值定义了正常、警告和警报范围的边界或区域。基于阈值的运行状况指示器有三种有效状态：“正常”、“警告”或“警报”。
- **基于状态**的指示器是有限状态集（包含一个对象的两种或更多种不同状态）度量，该状态集定义了数据库对象或资源的工作是否正常。其中一种状态表示情况正常，所有其他状态都被视为不正常。基于状态的运行状况指示器有两种有效状态：“正常”和“注意”。
- **基于集合状态**的指示器是数据库级度量，它们代表数据库中一个或多个对象的聚集状态。为该集合中的每个对象捕获数据，那些对象中最严重的情况以聚集状态表

示。如果该集合中有一个或多个对象处于要求发出警报的状态，运行状况指示器就会显示“注意”状态。基于集合状态的运行状况指示器有两种有效状态：“正常”和“注意”。

在实例级、数据库级、表空间级和表空间容器级都存在运行状况指示器。

可以通过“运行状况中心”、“Web 运行状况中心”、CLP 或 API 来访问运行状况监视器信息。可以通过这些工具来配置运行状况指示器。

当从正常状态切换到非正常状态，或者运行状况指示器值进入警告或警报区域（基于已定义的阈值边界），就会生成警报。有三种类型的警报：注意、警告和警报。

- 对于量度不同状态的运行状况指示器来说，如果注册了非正常状态，就会发出注意警报。
- 对于量度连续值范围的运行状况指示器来说，阈值定义了正常状态、警告状态和警报状态的边界或区域。例如，如果值进入定义警报区域的阈值范围，就会发出警报类型的警报以指示问题需要立即加以注意。

对于给定的运行状况指示器，运行状况监视器仅在特定报警情况第一次出现时发送通知并运行操作。如果运行状况指示器一直处于特定报警情况下，就不会发送更多通知，也不进一步运行操作。如果运行状况指示器更改了报警情况，或者回到正常状态并重新进入报警情况，就会发送新通知并运行操作。

下表显示了不同刷新时间间隔的运行状况指示器示例以及运行状况监视器对运行状况指示器状态的响应。此示例使用缺省警告阈值和警报阈值，它们分别是 80% 和 90%。

表 814. 不同刷新时间间隔的运行状况指示器条件

刷新时间间隔	ts.ts_util（表空间利用率）运行状况指示器值（%）	ts.ts_util 运行状况指示器状态	运行状况监视器响应
1	80	警告	发送警告通知，运行警告报警条件的操作
2	81	警告	不发送通知，不运行操作
3	75	正常	不发送通知，不运行操作
4	85	警告	发送警告通知，运行警告报警条件的操作
5	90	警报	发送警报通知，运行警报条件的操作

相关概念:

- 第 485 页的『运行状况监视器』

相关参考:

- 第 505 页的『运行状况指示器格式』
- 第 505 页的『运行状况指示器总结』

运行状况指示器处理周期

下图说明运行状况指示器的求值过程。每次经历运行状况指示器特定的刷新时间间隔时，就会运行这一组步骤。

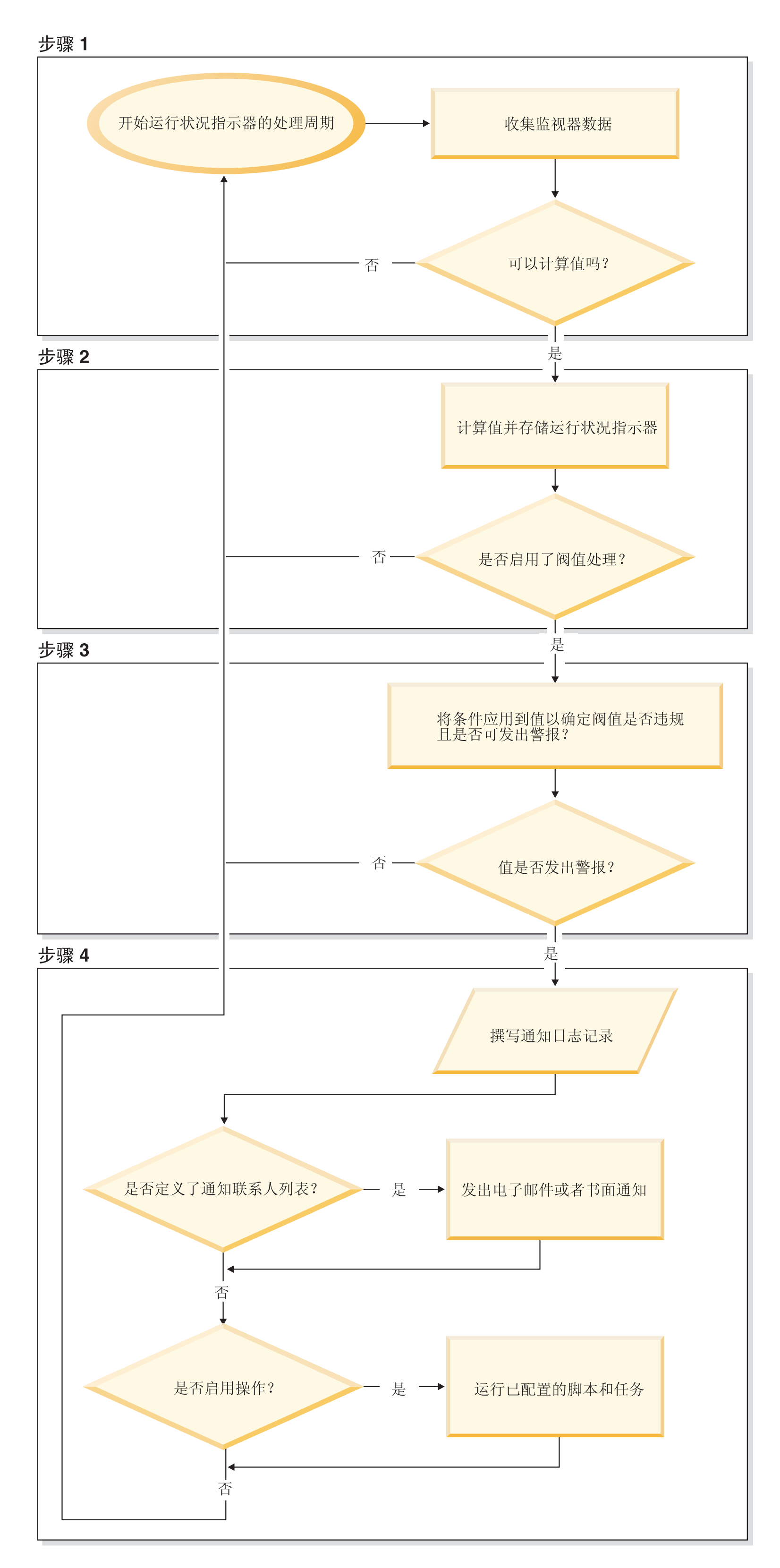


图 6. 运行状况指示器处理周期

注:

1. NOTIFYLEVEL 数据库管理器配置参数控制是将警报通知发送至 DB2 管理通知日志还是发送至所有定义的联系人。警报通知的最低严重性级别需要为 2。要发送警告和注意警报，最低严重性级别需要为 3。
2. 在 Windows 上迁移 DB2 数据库的版本 7 安装时，NOTIFYLEVEL 数据库管理器配置参数的值不会更新。

相关参考:

- 『notifylevel － 通知级别配置参数』（《*性能指南*》）
- 『UPDATE DATABASE MANAGER CONFIGURATION command』（*Command Reference*）

启用运行状况报警通知

要在生成警报时启用电子邮件或寻呼机通知，必须设置配置参数并指定联系人信息。

先决条件:

DB2 管理服务器（DAS）必须正在联系人列表所在的系统上运行。例如，如果 CONTACT_HOST 配置参数设置为远程系统，则为了获取要通知有关警报的联系人，DAS 必须在远程系统上运行。

过程:

要启用运行状况报警通知:

1. 指定 SMTP_SERVER 参数。

   DAS 配置参数 SMTP_SERVER 指定发送电子邮件和寻呼机通知消息时使用的邮件服务器的位置。如果安装 DB2 数据库的系统是作为未授权 SMTP 服务器启用的，则省略此步骤。
2. 指定 CONTACT_HOST 参数。

   DAS 配置参数 CONTACT_HOST 指定本地系统上所有实例的联系人列表的远程位置。通过设置此参数，可在多个系统间共享单个联系人列表。如果想要将联系人列表保留在安装 DB2 数据库的本地系统上，则省略此步骤。
3. 指定运行状况监视器通知的缺省联系人。

   为了能够在生成警报时从运行状况监视器获取电子邮件或寻呼机通知，必须指定缺省管理联系人。如果选择不提供此信息，则不会对报警条件发送通知消息。

   可在安装期间提供缺省管理联系人信息，也可以将此任务延迟至安装完成之后进行。

   如果选择延迟该任务或者想要将更多联系人或组添加至通知列表，则可以通过 CLP、C API 或“运行状况中心”来指定联系人:

   要使用 CLP 指定联系人:

   要将电子邮件联系人定义为运行状况监视器通知的缺省联系人，请发出以下命令:

```
DB2 ADD CONTACT contact_name TYPE EMAIL ADDRESS
   email_address DESCRIPTION 'Default Contact'

DB2 UPDATE NOTIFICATION LIST ADD CONTACT contact_name
```

有关完整的语句详细信息，请参阅 Command Reference。

要使用 C API 指定联系人:

下列 C 代码摘录说明如何定义运行状况通知联系人:

```
...
#include <db2ApiDf.h>

SQL_API_RC rc = 0;
struct db2AddContactData addContactData;
struct sqlca sqlca;

char* userid = "myuser";
char* password = "pwd";
char* contact = "DBA1";
char* email = "dba1@mail.com";
char* desc = "Default contact";

memset(&addContactData, '\0', sizeof(addContactData));
memset (&sqlca, '\0', sizeof(struct sqlca));
addContactData.piUserid = userid;
addContactData.piPassword = password;
addContactData.piName = contact;
addContactData.iType = DB2CONTACT_EMAIL;
addContactData.piAddress = email;
addContactData.iMaxPageLength = 0;
addContactData.piDescription = desc;

rc = db2AddContact(db2Version810, &addContactData, &sqlca);

if (rc == 0) {
   db2HealthNotificationListUpdate update;
   db2UpdateHealthNotificationListData data;
   db2ContactTypeData contact;

   contact.pName = contact;
   contact.contactType = DB2CONTACT_EMAIL;

   update.iUpdateType = DB2HEALTHNOTIFICATIONLIST_ADD;
   update.piContact = &contact;

   data.iNumUpdates = 1;
   data.piUpdates = &update;

   rc = db2UpdateHealthNotificationList (db2Version810, &data, &ca);
}
...
```

要使用“运行状况中心”指定联系人:

a. 右键单击想要对其定义运行状况通知列表的实例。

b. 单击**配置**，然后单击**警报通知**。“配置运行状况报警通知”窗口将打开。

c. 如果联系人未出现在窗口左边的**可用**列表中，则单击**管理联系人**。“联系人”窗口打开，并且带有预选系统名称。

d. 单击**添加联系人**。“添加联系人”窗口将打开。

e. 通过提供名称和电子邮件地址来定义联系人。如果指定电子邮件地址是寻呼机，则选择**将地址用于寻呼机**。

f. 单击**确定**。

g. 关闭“联系人”窗口并返回至“配置运行状况报警通知”窗口。新的联系人将出现在**可用联系人**列表中。

h. 通过单击向右方向按钮，将联系人移至**运行状况通知联系人列表**。

i. 单击**确定**以将该联系人包括在运行状况通知列表中。

建议 如果在通知时遇到困难，则选择运行状况通知联系人列表下面的**故障诊断**。“运行状况报警通知故障诊断”向导将打开。

相关参考:

- 『ADD CONTACT command using the ADMIN_CMD procedure』（*Administrative SQL Routines and Views*）
- 『UPDATE HEALTH NOTIFICATION CONTACT LIST command using the ADMIN_CMD procedure』（*Administrative SQL Routines and Views*）
- 『ADD CONTACT command』（*Command Reference*）
- 『UPDATE ADMIN CONFIGURATION command』（*Command Reference*）
- 『UPDATE HEALTH NOTIFICATION CONTACT LIST command』（*Command Reference*）

第 9 章 使用运行状况监视器

运行状况监视器

运行状况监视器捕获有关数据库管理器、数据库、表空间和表空间容器的信息。运行状况监视器根据从数据库系统监视元素、操作系统和 DB2 数据库检索到的信息计算运行状况指示器的各项指标。仅当数据库活动时，运行状况监视器才能对数据库及其对象计算运行状况指示器的各项指标。可通过使用 ACTIVATE DATABASE 命令启动数据库或保持与数据库的永久连接，以便让数据库保持活动状态。

运行状况监视器对每个运行状况指示器保留最大历史记录数。此历史记录存储在 <instance path>\hmonCache 目录中并且在停止运行状况监视器时除去。当达到最大记录数时，运行状况监视器将自动删除过时的历史记录。

运行状况监视器数据可通过运行状况快照访问。每个运行状况快照按最新刷新时间间隔报告每个运行状况指示器的状态。在检测现有数据库运行状况问题和预测数据库环境可能出现的不佳运行状况时，快照非常有用。可以使用 C 或 C++ 应用程序中的 API 或者图形管理工具来从 CLP 捕获运行状况快照。

运行状况监视需要实例连接。如果未使用 ATTACH TO 命令建立与实例的连接，则将创建与本地实例的缺省连接。

在分区数据库环境中，可在实例的任何分区上获取快照，或者使用单个实例连接获取全局快照。全局快照聚集在每个分区上收集到的数据并返回一组值。

使用说明:

DB2 数据库的所有版本都支持运行状况监视器。

要从“运行状况中心”启动或停止运行状况监视器，右键单击“运行状况中心”导航视图中的某个实例，然后选择“启动运行状况监视器”或“停止运行状况监视器”。

在 Windows 上，DB2 实例的服务需要在具有 SYSADM 权限的帐户下运行。可在 `db2icrt` 命令上使用“-u”选项，或在 Windows 上使用“服务”文件夹然后编辑“登录”属性以使用带有管理员特权的帐户。

在 Windows 上，运行状况监视器进程称为 DB2FMP。

DB2 管理服务器必须正在运行状况监视器所在的系统上运行，才能发送通知和运行警报操作。如果使用远程脚本、任务或联系人列表，则必须同时启动远程系统上的 DB2 管理服务器。

只有创建任务时才需要工具目录数据库。如果不对任何运行状况指示器使用警报任务操作，则运行状况监视器不需要工具目录数据库。

如果从更新版本的 DB2 数据库系统迁移回 DB2 UDB 版本 8.1，则所作的所有注册表更改将会丢失。注册表还原为版本 8.1 HealthRules.reg 文件，该文件包含使用更新注册表文件升级并启动之前存在的设置。

相关概念:

- 第 477 页的『运行状况监视器简介』

相关任务:

- 第 490 页的『通过客户机应用程序捕获数据库运行状况快照』
- 第 487 页的『使用 SQL 表函数捕获数据库运行状况快照』
- 第 488 页的『使用 CLP 捕获数据库运行状况快照』

相关参考:

- 第 495 页的『全局运行状况快照』
- 第 494 页的『运行状况监视器样本输出』
- 第 487 页的『运行状况监视器 SQL 表函数』

运行状况指示器数据

运行状况监视器对每个数据库分区上的每个运行状况指示器记录一组数据，包括:

- 运行状况指示器名称
- 值
- 评估时间戳记
- 警报状态
- 公式（如果适用的话）
- 附加信息（如果适用的话）
- 最多 10 个最新运行状况指示器求值历史记录。每个历史记录条目捕获导致当前运行状况指示器输出的下列运行状况指示器求值:
  - 值
  - 公式（如果适用的话）
  - 警报状态
  - 时间戳记

运行状况监视器还会跟踪实例、数据库和表空间级别的最高严重性警报状态。在每一级别，此运行状况指示器表示运行状况指示器在该级别或该级别之下的任何级别存在的最高严重性警报。例如，实例的最高严重性警报状态包括实例、该实例的任何数据库和每个数据库的任何表空间容器上的运行状况指示器。

相关参考:

- 第 517 页的『数据库最高严重性警报状态』
- 第 516 页的『实例最高严重性警报状态』

使用 SQL 表函数捕获数据库运行状况快照

可使用 SQL 表函数来捕获数据库运行状况快照。每个可用的运行状况快照表函数对应一个运行状况快照请求类型。

过程:

要使用 SQL 表函数来捕获数据库运行状况快照:

1. 标识计划使用的 SQL 表函数。

   SQL 表函数有两个输入参数:

   - VARCHAR(255), 用于数据库名称
   - INT, 用于分区号（0 到 999 之间的值）。输入与要监视的分区号相对应的整数。要捕获当前连接的分区的快照, 请输入值 -1。要捕获全局快照, 请输入值 -2。

   例外: 对于此规则, 数据库管理器快照 SQL 表函数是一个例外, 原因是它们只有一个参数。单一参数用于分区号。如果对数据库名称参数输入 NULL, 则监视器使用连接定义的数据库, 表函数就是通过该连接调用的。

2. 发出 SQL 语句。

   以下示例捕获当前连接的分区的基本运行状况快照, 以及连接定义的数据库上的基本运行状况快照（通过该连接调用此表函数）:

```
SELECT * FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1))
       as HEALTH_DB_INFO
```

   您也可以从返回的表中选择个别监视元素。返回的表中的每一列对应一个监视元素。相应地, 监视元素列名直接对应监视元素名称。以下语句仅返回数据库路径和服务器平台监视元素:

```
SELECT db_path, server_platform
       FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1 ) )
       as HEALTH_DB_INFO
```

相关概念:

- 第 485 页的『运行状况监视器』

相关参考:

- 第 529 页的『运行状况监视器接口』
- 第 487 页的『运行状况监视器 SQL 表函数』

运行状况监视器 SQL 表函数

下表列示所有快照表函数。每个表函数对应一个运行状况快照请求类型。

表 815. 快照监视器 SQL 表函数

监视器级别	SQL 表函数	返回的信息
数据库管理器	HEALTH_DBM_INFO	有关来自数据库管理器级别的运行状况快照的基本信息
数据库管理器	HEALTH_DBM_HI	来自数据库管理器级别的运行状况指示器信息

表 815. 快照监视器 SQL 表函数 (续)

监视器级别	SQL 表函数	返回的信息
数据库管理器	HEALTH_DBM_HI_HIS	来自数据库管理器级别的运行状况指示器历史记录信息
数据库	HEALTH_DB_INFO	有关来自数据库的运行状况快照的基本信息
数据库	HEALTH_DB_HI	来自数据库的运行状况指示器信息
数据库	HEALTH_DB_HI_HIS	来自数据库的运行状况指示器历史记录信息
数据库	HEALTH_DB_HIC	数据库的集合运行状况指示器的集合信息
数据库	HEALTH_DB_HIC_HIS	数据库的集合运行状况指示器的集合历史记录信息
表空间	HEALTH_TBS_INFO	有关数据库的表空间的运行状况快照的基本信息
表空间	HEALTH_TBS_HI	有关数据库的表空间的运行状况指示器信息
表空间	HEALTH_TBS_HI_HIS	有关数据库的表空间的运行状况指示器历史记录信息
表空间	HEALTH_CONT_INFO	有关数据库的容器的运行状况快照的基本信息
表空间	HEALTH_CONT_HI	有关数据库的容器的运行状况指示器信息
表空间	HEALTH_CONT_HI_HIS	有关数据库的容器的运行状况指示器历史记录信息

相关概念:

• 第 485 页的『运行状况监视器』

相关任务:

• 第 488 页的『使用 CLP 捕获数据库运行状况快照』

相关参考:

• 『Supported administrative SQL routines and views』（*Administrative SQL Routines and Views*）
• 第 529 页的『运行状况监视器接口』

使用 CLP 捕获数据库运行状况快照

可从 CLP 使用 GET HEALTH SNAPSHOT 命令来捕获运行状况快照。该命令语法支持检索运行状况监视器监视的不同对象类型的运行状况快照信息。

先决条件:

必须具有实例连接才能捕获运行状况快照。如果没有实例连接，则创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

过程:

要使用 CLP 捕获数据库运行状况快照

1. 从 CLP 发出带有期望参数的 GET HEALTH SNAPSHOT 命令。

在以下示例中，将在启动数据库管理器之后立即捕获数据库管理器级别运行状况快照。

```
db2 get health snapshot for dbm
```

2. 对于分区数据库系统，可为特定分区捕获专门的数据库快照，或者为所有分区捕获全局的数据库快照。要对特定分区（如分区号 2）上的数据库捕获运行状况快照，请发出以下命令:

```
db2 get health snapshot for db on sample at dbpartitionnum 2
```

要对所有分区上的所有应用程序捕获数据库快照，请发出以下命令:

```
db2 get health snapshot for db on sample global
```

以下命令捕获的运行状况快照带有附加详细信息，包括公式、附加信息和运行状况指示器历史记录:

```
db2 get health snapshot for db on sample show detail
```

3. 对于基于集合状态的运行状况指示器，可对所有集合对象捕获数据库快照，而不考虑这些对象的状态。常规 GET HEALTH SNAPSHOT FOR DB 命令返回所有集合对象，这些对象需要针对所有基于集合状态的运行状况指示器的警报。

要对列示了所有集合对象的数据库捕获运行状况快照，请发出以下命令:

```
db2 get health snapshot for db on sample with full collection
```

相关概念:

- 第 485 页的『运行状况监视器』

相关参考:

- 『GET HEALTH SNAPSHOT command』（*Command Reference*）
- 第 495 页的『全局运行状况快照』
- 第 489 页的『运行状况监视器 CLP 命令』
- 第 494 页的『运行状况监视器样本输出』

运行状况监视器 CLP 命令

下表列示了所有受支持的快照请求类型。

表 816. 快照监视器 CLP 命令

监视器级别	CLP 命令	返回的信息
数据库管理器	`get health snapshot for dbm`	数据库管理器级别信息。
数据库	`get health snapshot for all databases`	数据库级别信息。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	`get health snapshot for database on` *database-alias*	数据库级别信息。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	`get health snapshot for all on` *database-alias*	数据库、表空间和表空间容器信息。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
表空间	`get snapshot for tablespaces on` *database-alias*	连接到数据库的应用程序已访问的每个表空间的表空间级别信息。并且，返回的信息还包括该表空间中每个表空间容器的运行状况信息。

相关任务:

- 第 488 页的『使用 CLP 捕获数据库运行状况快照』

相关参考:

- 『GET HEALTH SNAPSHOT command』（*Command Reference*）
- 第 529 页的『运行状况监视器接口』

通过客户机应用程序捕获数据库运行状况快照

可使用 C 或 C++ 应用程序中的快照监视器 API 来捕获运行状况快照。可通过在 db2GetSnapshot API 中指定参数来访问若干不同运行状况快照请求类型。

先决条件:

必须连接至实例才能捕获运行状况快照。如果没有实例连接，则创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

过程:

1. 将 sqlmon.h 和 db2ApiDf.h DB2 库包括在节点中。这些库可在 sqllib\include 目录中找到。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

2. 将快照缓冲区单元大小设置为 50 KB。

   ```
   #define SNAPSHOT_BUFFER_UNIT_SZ 51200
   ```

3. 声明 sqlma、sqlca、sqlm_collected 和 db2GetSnapshotData 结构。

   ```
   struct sqlma *pRequestedDataGroups;
   struct sqlca sqlca;
   memset (&sqlca, '\0', sizeof(struct sqlca));
   struct sqlm_collected collectedData;
   memset (&sqlm_collected, '\0', sizeof(struct sqlm_collected));
   db2GetSnapshotData getSnapshotParam;
   memset(&db2GetSnapshotData, '\0', sizeof(db2GetSnapshotData));
   ```

4. 初始化指针以包含快照缓冲区并确定缓冲区大小。

   ```
   static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
   sqluint32 outputFormat;
   char *snapshotBuffer;
   ```

5. 初始化 sqlma 结构，并指定要捕获的快照属于数据库管理器级别信息。

   ```
   pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
   memset(&pRequestedDataGroups, '\0', sizeof(struct pRequestedDataGroups));
   pRequestedDataGroups->obj_num = 1;
   pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
   ```

6. 初始化缓冲区以容纳快照输出。

   ```
     snapshotBuffer = (char *)malloc(snapshotBufferSize);
   memset (&snapshotBuffer, '\0', sizeof(snapshotBuffer));
   ```

7. 使用快照请求类型（来自 sqlma 结构）、缓冲区信息和捕获快照所需的其他信息来填充 db2GetSnapshotData 结构。

   ```
   getSnapshotParam.piSqlmaData = pRequestedDataGroups;
   getSnapshotParam.poCollectedData = &collectedData;
   getSnapshotParam.poBuffer = snapshotBuffer;
   getSnapshotParam.iVersion = SQLM_DBMON_VERSION8;
   getSnapshotParam.iBufferSize = snapshotBufferSize;
   getSnapshotParam.iStoreResult = 0;
   ```

```
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_HEALTH;
```

8. 捕获运行状况快照。传递下列参数:
   - db2GetSnapshotData 结构，它包含捕获快照所需的信息
   - 对缓冲区的引用，快照输出将引导至该缓冲区。

   ```
   db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
   ```

9. 包括用于处理缓冲区溢出的逻辑。获取快照后，将检查 sqlcode 是否存在缓冲区溢出。如果发生了缓冲区溢出，则将清除并重新初始化缓冲区，然后再次获取快照。

   ```
   while (sqlca.sqlcode == 1606)
   {
   free(snapshotBuffer);
     snapshotBufferSize += SNAPSHOT_BUFFER_UNIT_SZ;
     snapshotBuffer = (char *)malloc(snapshotBufferSize);
     if (snapshotBuffer == NULL)
     {
       printf("\nMemory allocation error.\n");
       return;
     }

     getSnapshotParam.iBufferSize = snapshotBufferSize;
     getSnapshotParam.poBuffer = snapshotBuffer;
     db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
   }
   ```

10. 处理快照监视器数据流。参考遵循这些步骤之后的图形就会看到快照监视器数据流。
11. 清除缓冲区。

    ```
    free(snapshotBuffer);
    free(pRequestedDataGroups);
    ```

在使用 db2GetSnapshot API 捕获运行状况快照之后，该 API 以自描述数据流的形式返回运行状况快照输出。以下示例显示数据流结构:

```
SQLM_ELM_COLLECTED
  ├──► DB2
  │     └──► HEALTH_INDICATOR
  │           └──► HEALTH_INDICATOR_HIST ①
  ├──► DBASE
  │     └──► HEALTH_INDICATOR
  │           └──► HEALTH_INDICATOR_HIST ①
  └──► TABLESPACE_LIST
        └──► TABLESPACE
              ├──► HEALTH_INDICATOR
              │     └──► HEALTH_INDICATOR_HIST ①
              └──► NODEINFO ②
                    └──► TABLESPACE_CONTAINER
                          └──► HEALTH_INDICATOR
                                └──► HEALTH_INDICATOR_HIST ①
```

图 7. 运行状况快照自描述数据流

图注:

1. 只有在使用 SQLM_CLASS_HEALTH_WITH_DETAIL 快照类时才可用。
2. 只有在 DB2 企业服务器版中才可用。否则为表空间容器流。

下列层次结构显示运行状况快照自描述数据流中的特定元素。

SQLM_ELM_HI 中的各个元素的层次结构:

```
SQLM_ELM_HI
         SQLM_ELM_HI_ID
         SQLM_ELM_HI_VALUE
            SQLM_ELM_HI_TIMESTAMP
               SQLM_ELM_SECONDS
               SQLM_ELM_MICROSEC
   SQLM_ELM_HI_ALERT_STATE
```

SQLM_ELM_HI_HIST 中的各个元素的层次结构，仅对 SQLM_CLASS_HEALTH_WITH_DETAIL 快照类可用:

```
SQLM_ELM_HI_HIST
   SQLM_ELM_HI_FORMULA
   SQLM_ELM_HI_ADDITIONAL_INFO
      SQLM_ELM_HEALTH_INDICATOR_HIST
         SQLM_ELM_HI_ID
         SQLM_ELM_HI_VALUE
            SQLM_ELM_HI_TIMESTAMP
               SQLM_ELM_SECONDS
               SQLM_ELM_MICROSEC
         SQLM_ELM_HI_ALERT_STATE
         SQLM_ELM_HI_FORMULA
         SQLM_ELM_HI_ADDITIONAL_INFO
```

SQLM_ELM_OBJ_LIST 中的各个元素的层次结构:

```
SQLM_ELM_HI_OBJ_LIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_DETAIL
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

SQLM_ELM_OBJ_LIST_HIST 中的各个元素的层次结构，仅对 SQLM_CLASS_HEALTH_WITH_DETAIL 快照类可用:

```
SQLM_ELM_HI_OBJ_LIST_HIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

相关概念:

- 第 485 页的『运行状况监视器』
- 第 49 页的『快照监视器自描述数据流』
- 第 7 页的『系统监视器输出: 自描述数据流』

相关参考:

- 『db2GetSnapshot API - Get a snapshot of the database manager operational status』（*Administrative API Reference*）
- 『db2GetSnapshotSize API - Estimate the output buffer size required for the db2GetSnapshot API』（*Administrative API Reference*）
- 第 493 页的『运行状况监视器 API 请求类型』

运行状况监视器 API 请求类型

下表列示了所有受支持的快照请求类型。

表 817. 快照监视器 API 请求类型

监视器级别	API 请求类型	返回的信息
数据库管理器	SQLMA_DB2	数据库管理器级别信息。
数据库	SQLMA_DBASE_ALL	数据库级别信息。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE	数据库级别信息。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
表空间	SQLMA_DBASE_TABLESPACES	连接到数据库的应用程序已访问的每个表空间的表空间级别信息。并且，返回的信息还包括该表空间中每个表空间容器的运行状况信息。

相关概念:

- 第 49 页的『快照监视器自描述数据流』

相关任务:

- 第 490 页的『通过客户机应用程序捕获数据库运行状况快照』

相关参考:

- 『db2GetSnapshot API - Get a snapshot of the database manager operational status』(*Administrative API Reference*)
- 『db2GetSnapshotSize API - Estimate the output buffer size required for the db2GetSnapshot API』(*Administrative API Reference*)

运行状况监视器样本输出

下列示例显示使用 CLP 获取的运行状况快照及其相应输出，并说明运行状况监视器的特征。这些示例的目的是启动数据库管理器后立即检查整体运行状况状态。

示例 1 过程:

1. 使用 GET HEALTH SNAPSHOT 命令获取数据库管理器快照:

   ```
   db2 get health snapshot for dbm
   ```

   在从 CLP 发出 GET HEALTH SNAPSHOT 命令后，快照输出将引导至屏幕。

   ```
   节点名                                    =
   节点类型                                  = 带有本地和远程客户机的数据库服务器
   实例名                                    = DB2
   快照时间戳记                              = 11-07-2002 12:43:23.613425

   DB2 实例中的数据库分区数                  = 1
   启动数据库管理器时间戳记                  = 11-07-2002 12:43:18.000108
   实例最高严重性警报状态                    = 尚未求值

   运行状况指示器:

      尚未求值
   ```

2. 分析输出。

   从此运行状况快照中，可以看到实例最高严重性警报状态为“Not yet evaluated”。实例处于此状态的原因是运行状况监视器刚刚启动，还没有计算任何运行状况指示器的各项指标。

   如果实例最高严重性警报状态未更改，则:

   - 检查 HEALTH_MON 数据库管理器配置参数的值以确定运行状况监视器是否已启动。
   - 如果 `HEALTH_MON=OFF`，则表示运行状况监视器未启动。要启动运行状况监视器，请发出 `UPDATE DBM CFG USING HEALTH_MON ON` 命令。
   - 如果 `HEALTH_MON=ON`，则连接至实例以激活运行状况监视器。如果存在实例连接，则可能不能将运行状况监视器装入到内存中。

使用 CLP 获取数据库运行状况快照的另一示例如下所述。

示例 2 先决条件:

- 数据库连接必须存在。
- 数据库必须处于停顿状态。

示例 2 过程:

1. 使用 GET HEALTH SNAPSHOT 命令获取数据库管理器快照:

   ```
   db2 get health snapshot for db on sample
   ```

   在从 CLP 发出 GET HEALTH SNAPSHOT 命令后，快照输出将引导至屏幕。

   ```
                Database Health Snapshot

   快照时间戳记                              = 12-09-2002 11:44:37.793184

   数据库名称                                = SAMPLE
   数据库路径                                = E:\DB2\NODE0000\SQL00002\
   输入数据库别名                            = SAMPLE
   在数据库服务器上运行的操作系统            = NT
   数据库的位置                              = 本地
   数据库最高严重性警报状态                  = 注意

   运行状况指示器:

    ...
       指示器名称                                = db.log_util
          值                                     = 60
          单位                                   = %
          求值时间戳记                           = 12-09-2002 11:44:00.095000
          警报状态                               = 正常

       指示器名称                                = db.db_op_status
          值                                     = 2
          求值时间戳记                           = 12-09-2002 11:44:00.095000
          警报状态                               = 注意
   ```

2. 分析输出。

   此运行状况快照显示 *db.db_op_status* 运行状况指示器上存在注意警报。值 2 指示数据库处于停顿状态。

相关概念:

- 第 485 页的『运行状况监视器』

相关参考:

- 第 489 页的『运行状况监视器 CLP 命令』

全局运行状况快照

在分区数据库系统上，可获取当前分区、指定分区或所有分区的运行状况快照。对分区数据库的所有分区获取全局运行状况快照时，会尽可能地先聚集数据，然后返回结果。

运行状况指示器的聚集警报状态相当于所有数据库分区上的最高严重性警报状态。不能聚集各个数据库分区上的附加信息和历史记录数据，因此不会提供它们。运行状况指示器的余下数据的聚集将在下表中作详细描述。

表 818. 运行状况指示器值、时间戳记和公式数据的聚集

运行状况指示器	聚集详细信息
• db2.db2_op_status • db2.sort_privmem_util • db2.mon_heap_util • db.db_op_status • db.sort_shrmem_util • db.spilled_sorts • db.log_util • db.log_fs_util • db.locklist_util • db.apps_waiting_locks • db.db_heap_util • db.db_backup_req • ts.ts_util	从包含最高值的分区获取运行状况指示器值。 从同一分区获取求值时间戳记和公式。
• db.max_sort_shrmem_util • db.pkgcache_hitratio • db.catcache_hitratio • db.shrworkspace_hitratio	从包含最低值的分区获取运行状况指示器值。 从同一分区获取求值时间戳记和公式。
• db.deadlock_rate • db.lock_escal_rate	运行状况指示器值是所有数据库分区上的值的总和。 不能聚集求值时间戳记和公式，所以不提供它们。
• ts.ts_op_status • tsc.tscont_op_status • tsc.tscont_util	不聚集这些运行状况指示器。
• db.hadr_op_status • db.hadr_log_delay	多分区数据库不支持这些运行状况指示器。
• db.tb_reorg_req • db.tb_runstats_req • db.fed_nicknames_op_status • db.fed_servers_op_status	仅在一个分区上对此运行状况指示器求值，所以不需要聚集。从对运行状况指示器求值的分区返回数据。

注: 对单个分区对象获取全局快照时，输出将包括所有属性，这是因为没有要聚集的分区。

相关概念:

• 第 485 页的『运行状况监视器』

相关任务:

• 第 488 页的『使用 CLP 捕获数据库运行状况快照』
• 第 487 页的『使用 SQL 表函数捕获数据库运行状况快照』
• 第 490 页的『通过客户机应用程序捕获数据库运行状况快照』

运行状况监视器的图形工具

运行状况中心

“运行状况中心”是一个图形管理工具，用于支持按异常情况进行管理。对于在客户机上编目的所有 Windows、Linux 及 UNIX 实例和数据库，“运行状况中心”提供:

- 用于查看所有实例及其数据库的累积警报状态的中央位置
- 用于查看实例和数据库及其子对象的当前警报的图形界面
- 用于访问当前警报的详细信息和建议解决措施的图形界面

要从命令行启动“运行状况中心”，请输入 db2hc 命令。

在 Windows 上，还可通过单击**开始 → 程序 → IBM DB2 →<DB2 copy name>→ 监视工具 → 运行状况中心**来从“开始”菜单启动“运行状况中心”。

“运行状况中心”的左边面板中是导航树，右边面板中是“警报”视图。“导航”视图的内容将根据在“导航”视图顶部选择的切换按钮进行过滤。

“运行状况中心”打开时选择了**处于任何警报状态的对象**切换按钮，这有助于标识带有应该解决的当前警报的实例。选择**所有对象**切换按钮后，将显示在客户机上编目的所有的Windows、Linux 和 UNIX 实例及它们各自的状态。没有图标的实例表示没有正在运行的运行状况监视器，或者是版本 8 以前的实例（缺少对运行状况监视器功能的支持）。

选择实例时，“运行状况中心”将从运行状况监视器请求所选实例的状态。“警报”视图将填写实例、该实例的任何数据库和每个数据库的任何表空间和表空间容器上的所有当前警报。如果在“导航”视图中展开实例并选择子代数据库对象，则“警报”视图将被限制为仅显示所选数据库及其任意表空间或表空间容器的警报。

刷新图标在“运行状况中心”的右上角。单击刷新图标以便立即刷新，或者设置特定刷新时间间隔，这将导致“运行状况中心”在服务器上查询运行状况监视器的当前状态。此查询不会导致运行状况监视器刷新运行状况指示器求值。每个运行状况指示器都具有定义的刷新时间间隔。仅当经过刷新时间间隔时，才会重新对运行状况指示器的警报状态求值。每次定时刷新或请求刷新“运行状况中心”时，仅显示运行状况指示器的当前状态。

“警报”视图具有一个功能，它可以定义带有特定定制列和排序顺序的定制视图。“运行状况中心”中有六种预定义视图可用来定制个人命名和分类方案。可通过使用窗口底部的工具栏或选择**视图**菜单中的**已保存**视图来选择预定义视图。要定义您自己的定制视图，单击窗口底部工具栏上的**视图**按钮，或者使用**视图**菜单。在下一次调用“运行状况中心”时，记住为显示“警报”视图中的数据而选择的视图。

要获取警报的详细信息，在“警报”视图中选择警报行。使用**已选择**菜单，或右键单击该行并选择**显示详细信息**。详细信息窗口显示警报的详细信息，包括发生警报的对象和分区、公式（如果适用的话）和运行状况指示器的值。

对于基于阈值的运行状况指示器，将显示用于确定报警条件的阈值。详细信息窗口还将显示运行状况指示器的附加信息。此信息可能包括配置参数的值或提供警报上下文的其他监视器数据。将显示对运行状况指示器的描述，包括运行状况指示器的用途及其作为重要度量属性的原因。

对于基于集合状态的运行状况指示器，集合对象列表将显示在**运行状况指示器警报状态**表的“对象”中。表中将提供对象名、状态、时间戳记和详细信息。

详细信息页上提供了**查看历史记录**按钮。将从运行状况指示器求值的第二次刷新开始存储运行状况指示器的历史记录。只有在存储历史记录后才会在“运行状况中心”的“查看历史记录”对话框中显示内容。对于基于集合状态的运行状况指示器，可通过单击历史记录窗口中的**查看集合历史记录**按钮来查看集合对象的历史记录。

运行状况中心状态信标

“运行状况中心状态信标”是可在 DB2 管理工具中启用的可视指示器。当“运行状况中心”未打开并且您在使用其他 DB2 管理工具时，该信标将向您通知当前警报。该信标用于因为报警条件而提示用户打开“运行状况中心”。

“运行状况中心状态信标”有两种不同的通知方法。一种通知方法使用弹出消息。另一种通知方法使用显示在打开窗口的状态行右边部分的图形信标。图形信标包括一个按钮，单击该按钮可访问“运行状况中心”。

两种信标通知方法都可以通过“工具设置”对话框启用。“通过弹出消息通知”方法控制弹出消息通知，而“通过状态行通知”方法控制可视信标。

Web 运行状况中心

“Web 运行状况中心”可通过 Web 浏览器或基于 Web 的个人数字助理（PDA）访问。提供给用户的界面随要使用的介质的不同而变化，但内容是相同的。“Web 运行状况中心”供通常使用完整“运行状况中心”但目前离开其通用访问点的用户使用。

“Web 运行状况中心”允许您查看特定实例及其所有子代对象的运行状况信息。不能通过“Web 运行状况中心”选择特定于数据库的上下文。

使用“Web 运行状况中心”的第一步是登录至安装时设置的 DB2 Web 工具站点上的应用程序。从首页单击“运行状况中心”选项卡（或 PDA 上的链接）。首先，将提示您选择系统并提供用户标识和密码进行认证。接着，必须选择想要查看其运行状况状态的实例。

DB2 Web 工具可自动使用 DB2 发现来编目网络上的系统、实例和数据库。如果选择通过在 web.xml 中将标志设置为 false 以禁用自动编目选项，则必须手工编目应用程序服务器上的系统、实例和数据库。

选择实例时，“Web 运行状况中心”将打开“当前警报”页。此页列示所选实例、该实例的任何已编目数据库及这些数据库中的任何表空间和表空间容器上当前存在的所有警报。对于“运行状况中心”，对警报的完整详细信息的描述可通过单击警报获取。

PDA 中的“Web 运行状况中心”在“描述”视图中显示运行状况指示器的详细信息，该视图是“当前警报”视图中可访问的第一个屏幕。可通过“描述”视图顶部的链接访问运行状况指示器建议和历史记录。

运行状况中心概述

使用“运行状况中心”来分析和改进 DB2 的运行状况。以下是定义使得 DB2 运行状况良好的条件的示例:

- 有足够的资源（如可用内存、表空间容器或记录存储器）来完成任务。
- 资源使用效率较高。

- 执行任务的时间在可接受范围以内，或者任务的执行不会导致性能显著下降。
- 资源或数据库对象不会无限期处于不可用状态。

还可从“运行状况中心”打开其他中心和工具，以帮助您调查和维护 DB2 的运行状况。

要在 Intel 平台上打开“运行状况中心”，从**开始**菜单单击**开始** → **程序** → **IBM DB2** → **监视工具** → **运行状况中心**。

要在 Intel 平台上使用命令行打开“运行状况中心”，运行以下命令:

```
db2hc
```

运行状况中心中的任务:

以下列表对可以在“运行状况中心”中执行的一些关键任务进行了分类:

- 启用运行状况报警通知
  - 指定联系人设置和通知配置参数
  - 故障诊断运行状况报警通知
- 使用运行状况中心配置运行状况指示器
  - 启用和禁用运行状况指示器求值
  - 更改警报阈值和灵敏度设置
  - 发生警报时运行任务和脚本
- 使用运行状况中心解决警报
  - 使用建议顾问程序来选择和实施建议

运行状况中心界面:

“运行状况中心”界面具有下列元素，可以帮助您确定和解决与系统的整体运行状况有关的问题。

运行状况中心菜单栏



使用菜单栏在“运行状况中心”中处理对象，打开其他管理中心和工具和访问联机帮助。

“运行状况中心”菜单栏包含下列菜单:

运行状况中心工具栏



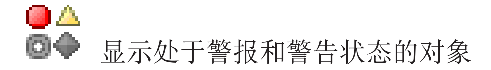
使用菜单栏下面的工具栏图标来访问其他中心和工具，以及刷新“运行状况中心”的内容视图。

切换按钮



使用切换按钮来选择“导航”视图中出现的警报状态。每个按钮对应一个数据库对象在视图中显示时需要的最低警报严重性。选择不同的按钮只会影响显示内容，不会影响该对象本身。


显示处于警报状态的对象


显示处于警报和警告状态的对象

显示处于以下任一警报状态的对象：警报、警告、注意、正常和未监视。


显示所有对象

导航视图



使用“导航”视图来显示和处理实例和数据库对象。当选择“导航”视图中的对象时，该对象及其所有子代的当前警报将显示在“警报”视图中。要在“导航”视图显示该对象之前更改该对象必须具有的级别，在“导航”视图中右键单击远离列示对象的位置。这将会打开警报级别的弹出菜单。选择想要显示的警报级别。还可通过单击切换按钮来选择要显示的警报级别。

警报视图

使用“警报”视图来显示和处理当前警报。“警报”视图显示在“导航”视图中选择的对象及其子代数据库对象当前存在的警报。例如，如果选择某个实例，则会显示该实例及其所有数据库和表空间的警报。如果选择某个数据库，则会显示该数据库及其所有表空间的警报。在“警报”视图中选择并右键单击一个或多个警报以对其调用操作。

警报视图工具栏



使用“警报”视图下面的工具栏来调整“警报”视图中的警报视图以满足您的需要。

用键盘访问定制控件:

可使用键盘来访问用户界面上的控件。

要了解更多信息，请参阅键盘快捷键和加速键（所有中心）。

第 4 部分 运行状况监视器参考

第 10 章 运行状况监视器逻辑数据组

运行状况监视器接口至逻辑数据组的映射

下表列示了所有受支持的运行状况快照请求类型。

表 819. 运行状况监视器接口至逻辑数据组的映射

API 请求类型	CLP 命令	SQL 表函数	逻辑数据组
SQLMA_DB2	get health snapshot for dbm	HEALTH_DBM_INFO	db2
		HEALTH_DBM_HI	health_indicator
	get health snapshot for dbm show detail	HEALTH_DBM_HI_HIS	health_indicator_history
SQLMA_DBASE	get health snapshot for database on *dbname*	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for database on *dbname* show detail	HEALTH_DB_HI_HIS	health_indicator_history
SQLMA_DBASE with SQLM_HMON_OPT_COLL_FULL in the agent_id	get health snapshot for database on *dbname* with full collection	HEALTH_DB_HIC	health_indicator，hi_obj_list
	get health snapshot for database on *dbname* show detail with full collection	HEALTH_DB_HIC_HIST	health_indicator_history，hi_obj_list
SQLMA_DBASE_ALL	get health snapshot for all databases	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for all databases show detail	HEALTH_DB_HI_HIS	health_indicator_history
SQLMA_DBASE_TABLESPACES	get health snapshot for tablespaces on *dbname*	HEALTH_TS_INFO	tablespace
		HEALTH_TS_HI	health_indicator
		HEALTH_CONT_INFO	tablespace_container
		HEALTH_CONT_HI	health_indicator
	get health snapshot for tablespaces on *dbname* show detail	HEALTH_TS_HI_HIS	health_indicator_history
		HEALTH_CONT_HI_HIS	health_indicator_history

下图显示逻辑数据分组在运行状况快照数据流中可能出现的顺序。

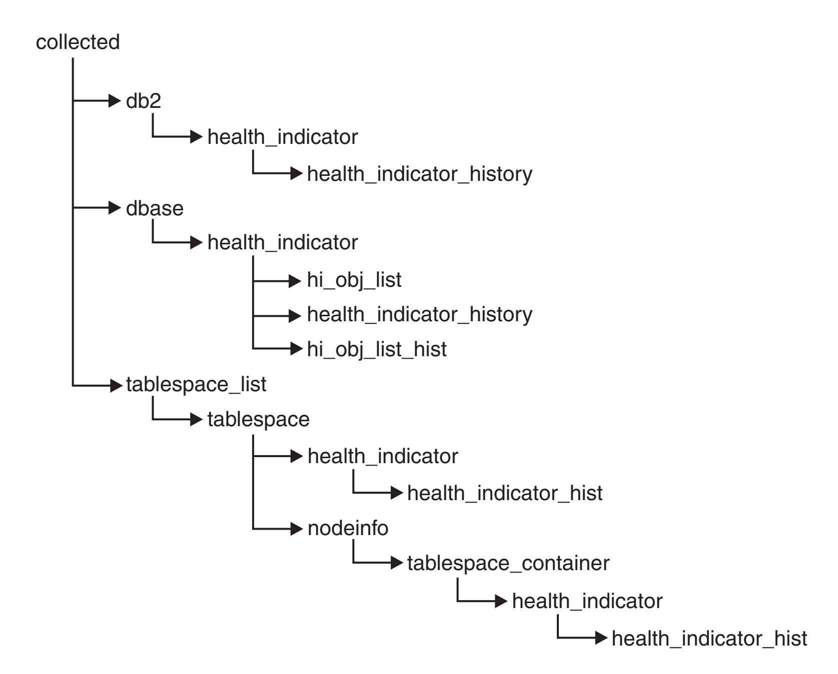


图 8. 运行状况快照逻辑数据分组

相关参考:

- 『GET HEALTH SNAPSHOT command』(*Command Reference*)
- 第 529 页的『运行状况监视器接口』

第 11 章 运行状况指示器

运行状况指示器格式

运行状况指示器的记录是以如下标准格式描述的:

标识
运行状况指示器的名称。此标识用于 CLP 中的配置。

运行状况监视器级别
运行状况监视器捕获运行状况指示器的级别。

类别
运行状况指示器的类别。

类型
运行状况指示器的类型。该类型有四个可能的值:

- 基于阈值上限，其中警报级数为：正常，警告，警报
- 基于阈值下限
- 基于状态，其中一种状态表示情况正常，所有其他状态都被视为不正常
- 基于集合状态，其中状态基于集合中对象的状态聚集

单位
运行状况指示器中度量的数据单位，如百分比。此项对基于状态或集合状态的运行状况指示器不适用。

描述
对运行状况指示器所收集数据的描述。

相关概念:

- 第 3 页的『数据库系统监视器』
- 第 3 页的『数据库系统监视器数据结构』

相关参考:

- 第 477 页的『运行状况指示器』

运行状况指示器总结

下表按类别分组列示所有运行状况指示器。

表 820. 数据库自动存储器利用率运行状况指示器

名称	标识	其他信息
数据库自动存储器利用率	db.auto_storage_util	第 508 页的『db.auto_storage_util － 数据库自动存储器利用率运行状况指示器』

表 821. 表空间存储器运行状况指示器

名称	标识	其他信息
表空间自动调整大小状态	ts.ts_auto_resize_status	第 509 页的『ts.ts_auto_resize_status － 表空间自动调整大小状态运行状况指示器』
自动调整大小状态表空间利用率	ts.ts_util_auto_resize	第 509 页的『ts.ts_util_auto_resize － 自动调整大小表空间利用率运行状况指示器』
表空间利用率	ts.ts_util	第 510 页的『ts.ts_util － 表空间利用率』

表 821. 表空间存储器运行状况指示器（续）

名称	标识	其他信息
表空间容器利用率	tsc.tscont_util	第 511 页的『tsc.tscont_util － 表空间容器利用率』
表空间运作状态	ts.ts_op_status	第 512 页的『ts.ts_op_status － 表空间操作状态』
表空间容器运作状态	tsc.tscont_op_status	第 512 页的『tsc.tscont_op_status － 表空间容器操作状态』
表空间自动调整大小状态	ts.ts_auto_resize_status	第 509 页的『ts.ts_auto_resize_status － 表空间自动调整大小状态运行状况指示器』

表 822. 排序运行状况指示器

名称	标识	其他信息
专用排序内存利用率	db2.sort_privmem_util	第 513 页的『db2.sort_privmem_util － 专用排序内存利用率』
共享排序内存利用率	db.sort_shrmem_util	第 513 页的『db.sort_shrmem_util － 共享排序内存利用率』
溢出排序百分比	db.spilled_sorts	第 514 页的『db.spilled_sorts － 溢出排序的百分比』
长期共享排序内存利用率	db.max_sort_shrmem_util	第 515 页的『db.max_sort_shrmem_util － 长期共享排序内存利用率』

表 823. 数据库管理器运行状况指示器

名称	标识	其他信息
实例运作状态	db2.db2_op_status	第 516 页的『db2.db2_op_status － 实例工作状态』
实例最高严重性警报状态	–	第 516 页的『实例最高严重性警报状态』

表 824. 数据库运行状况指示器

名称	标识	其他信息
数据库运作状态	db.db_op_status	第 517 页的『db.db_op_status － 数据库操作状态』
数据库最高严重性警报状态	–	第 517 页的『数据库最高严重性警报状态』

表 825. 维护运行状况指示器

名称	标识	其他信息
需要重组	db.tb_reorg_req	第 517 页的『db.tb_reorg_req － 需要重组』
必需的统计信息收集运行状况指示器	db.tb_runstats_req	第 518 页的『db.tb_runstats_req － 需要收集统计信息』
必需的数据库备份	db.db_backup_req	第 518 页的『db.db_backup_req － 需要数据库备份』

表 826. 高可用性灾难恢复运行状况指示器

名称	标识	其他信息
HADR 运作状态运行状况指示器	db.hadr_op_status	第 519 页的『db.hadr_op_status － HADR 操作状态』
HADR 日志延迟运行状况指示器	db.hadr_delay	第 519 页的『db.hadr_delay － HADR 日志延迟』

表 827. 日志记录运行状况指示器

名称	标识	其他信息
日志利用率	db.log_util	第 520 页的『db.log_util － 日志利用率』
日志文件系统利用率	db.log_fs_util	第 520 页的『db.log_fs_util － 日志文件系统利用率』

表 828. 应用程序并发性运行状况指示器

名称	标识	其他信息
死锁率	db.deadlock_rate	第 521 页的『db.deadlock_rate － 死锁率』

表 828. 应用程序并发性运行状况指示器（续）

名称	标识	其他信息
锁定列表利用率	db.locklist_util	第 522 页的『db.locklist_util － 锁定列表利用率』
锁定升级率	db.lock_escal_rate	第 522 页的『db.lock_escal_rate － 锁定升级率』
等待锁定的应用程序的百分比	db.apps_waiting_locks	第 523 页的『db.apps_waiting_locks － 等待锁定的应用程序所占的百分比』

表 829. 程序包和目录高速缓存，以及工作空间运行状况指示器

名称	标识	其他信息
目录高速缓存命中率	db.catcache_hitratio	第 524 页的『db.catcache_hitratio － 目录高速缓存命中率』
程序包高速缓存命中率	db.pkgcache_hitratio	第 524 页的『db.pkgcache_hitratio － 程序包高速缓存命中率』
共享工作空间命中率	db.shrworkspace_hitratio	第 525 页的『db.shrworkspace_hitratio － 共享工作空间命中率』

表 830. 内存运行状况指示器

名称	标识	其他信息
监视器堆利用率	db2.mon_heap_util	第 525 页的『db2.mon_heap_util － 监视器堆利用率』
数据库堆利用率	db.db_heap_util	第 526 页的『db.db_heap_util － 数据库堆利用率』

表 831. 联合运行状况指示器

名称	标识	其他信息
昵称状态	db.fed_nicknames_op_status	第 526 页的『db.fed_nicknames_op_status － 昵称状态』
数据源服务器状态	db.fed_servers_op_status	第 527 页的『db.fed_servers_op_status － 数据源服务器状态』

相关参考:

- 第 477 页的『运行状况指示器』

DMS 表空间的运行状况指示器

此表描述根据表空间的特征来看与 DMS 表空间有关的表空间运行状况指示器:

表 832. DMS 表空间的相关表空间运行状况指示器

表空间特征	定义的最大表空间大小	未定义的最大表空间大小
Automatic resize enabled = Yes	ts.ts_util_auto_resize – 跟踪相对于定义的最大值而言使用的空间在表空间中所占的百分比。警报指示表空间很快就会变满并且需要您的干预。只要将最大大小设置为合理的值（即最大大小指定的空间量存在），它就是此配置最重要的运行状况指示器。 ts.ts_util – 跟踪当前分配的表空间存储器使用情况。警报可能不需要您的干预就能够解决任何问题，原因是表空间在变满时将尝试增加大小。 ts.ts_auto_resize_status – 跟踪调整大小尝试的运行状况。警报指示表空间调整大小失败（即表空间已满）。	ts.ts_util_auto_resize – 不适用。未对表空间大小指定上边界。 ts.ts_util – 跟踪当前分配的表空间存储器使用情况。警报可能不需要您的干预就能够解决任何问题，原因是表空间将尝试增加大小。 ts.ts_auto_resize_status – 跟踪调整大小尝试的运行状况。警报指示表空间调整大小失败（即表空间已满）。 **注:** 如果使用自动存储器定义了 DMS 表空间并且未指定最大大小，则还应注意 db.auto_storage_util 运行状况指示器。此运行状况指示器跟踪与数据库存储器路径相关联的空间的利用率。当此空间变满时，表空间将无法增长。这可能导致出现表空间已满的情况。
Automatic resize enabled = No	配置无效。最大表空间大小仅对能够自动调整大小的表空间有效。	ts.ts_util_auto_resize – 不适用。表空间将不尝试调整大小。 ts.ts_util – 跟踪当前分配的表空间存储器使用情况。警报指示表空间已满情况并且需要您立即干预。表空间自身不会尝试调整大小。 ts.ts_auto_resize_status – 不适用。表空间将不尝试调整大小。

数据库存储器运行状况指示器

db.auto_storage_util – 数据库自动存储器利用率运行状况指示器

标识 db.auto_storage_util

运行状况监视器级别 数据库

类别 数据库

类型 基于阈值上限

单位 百分比

描述 此运行状况指示器跟踪定义的数据库存储器路径的存储器消耗情况。创建自动存储器表空间时，将在数据库存储器路径上为这些表空间自动分配容器。如果定义了数据库存储器路径的任何文件系统上没有更多空间，则自动存储器表空间将无法增加大小，从而可能会变满。

使用以下公式计算指示器:

```
(db.auto_storage_used / db.auto_storage_total) * 100
```

其中 *db.auto_storage_used* 和 *db.auto_storage_total* 分别是数据库存储器路径列表中标识的所有物理文件系统上的已使用空间总和及总空间。

数据库自动存储器路径利用率以数据库存储器路径文件系统上消耗空间的百分比来度量，其中高百分比指示未达到此指示器的最优运行状况。

附加信息中对离空间已满所余时间的计算可以预测到消耗所有可用空间所余的时间。

相关概念:

- 『自动存储器数据库』（*《管理指南：实施》*）

相关参考:

- 第 517 页的『数据库最高严重性警报状态』

表空间存储器运行状况指示器

ts.ts_auto_resize_status － 表空间自动调整大小状态运行状况指示器

标识	ts.ts_auto_resize_status
运行状况监视器级别	表空间
类别	表空间存储器
类型	基于状态
单位	不适用

描述 此运行状况指示器标识对能够自动调整大小的 DMS 表空间执行表空间调整大小操作是否成功。当能够自动调整大小的 DMS 表空间未能增加大小时，它会很快变满。此情况可能是因为定义了表空间容器的文件系统上缺少可用空间造成的，也可能是表空间自动调整大小设置造成的。例如，可能已经达到定义的最大大小，或者增加量可能设置得太高而导致余下可用空间无法容纳。

相关参考:

- 第 508 页的『db.auto_storage_util － 数据库自动存储器利用率运行状况指示器』
- 第 507 页的『DMS 表空间的运行状况指示器』
- 第 320 页的『tablespace_last_resize_failed － 上一次调整大小尝试失败』
- 第 510 页的『ts.ts_util － 表空间利用率』
- 第 512 页的『tsc.tscont_op_status － 表空间容器操作状态』
- 第 511 页的『tsc.tscont_util － 表空间容器利用率』

ts.ts_util_auto_resize － 自动调整大小表空间利用率运行状况指示器

标识	ts.ts_util_auto_resize
运行状况监视器级别	表空间
类别	表空间存储器

类型 基于阈值上限

单位 百分比

描述 此运行状况指示器跟踪每个 DMS 表空间的存储器消耗情况，这些 DMS 表空间已经定义了最大大小，并且可以自动调整大小。达到最大大小时，则认为 DMS 表空间已满。

使用以下公式计算指示器:

```
((ts.used * ts.page_size) / ts.max_size) * 100
```

其中 *ts.used*、*ts.page_size* 和 *ts.max_size* 分别是系统监视器数据元素“表空间中使用的页数”、“表空间的页大小”和“最大表空间大小”。

自动调整大小 DMS 表空间利用率是用消耗的最大表空间存储器所占的百分比度量的。高百分比指示表空间接近已满程度。此指示符的附加信息中包括的短期增长率和长期增长率可用来确定，当前增长率是短期畸变还是与长期增长一致。

附加信息中对离空间已满所余时间的计算可以预测达到最大大小所余的时间。

相关参考:

- 第 507 页的『DMS 表空间的运行状况指示器』
- 第 318 页的『tablespace_current_size － 当前表空间大小』
- 第 319 页的『tablespace_max_size － 最大表空间大小』
- 第 510 页的『ts.ts_util － 表空间利用率』
- 第 512 页的『tsc.tscont_op_status － 表空间容器操作状态』
- 第 511 页的『tsc.tscont_util － 表空间容器利用率』
- 第 509 页的『ts.ts_auto_resize_status － 表空间自动调整大小状态运行状况指示器』

ts.ts_util － 表空间利用率

标识 ts.ts_util

运行状况监视器级别 表空间

类别 表空间存储器

类型 基于阈值上限

单位 百分比

描述 此运行状况指示器跟踪每个 DMS 表空间的存储器消耗情况。

当所有容器已满时，则认为 DMS 表空间已满。

如果在表空间上启用自动调整大小，则不会评估此运行状况指示器。相反，对于表空间存储器监视，数据库自动存储器利用率 *db.auto_storage_util* 运行状况指示器和表空间自动调整大小状态 *ts.ts_auto_resize_status* 运行状况指示器是相关的。如果对此表空间定义了最大大小，则自动调整大小表空间利用率 *ts.ts_util_auto_resize* 运行状况指示器还将可用。如果需要，还可从 TBSP_UTILIZATION 管理视图的 TBSP_UTILIZATION_PERCENT 列检索表空间利用率百分比。

使用以下公式计算指示器:

```
(ts.used / ts.useable) * 100
```

其中 *ts.used* 和 *ts.useable* 分别为系统监视元素*表空间中使用的页数*和*表空间可用的页数*。

表空间利用率以消耗空间的百分比来度量，其中高百分比指示未达到此指示器的最优运行状况。

此指示符的附加信息中包括的短期增长率和长期增长率可用来确定，当前增长率是短期畸变还是与长期增长一致。

附加信息中对离空间已满所余时间的计算可以预测到消耗所有可用空间所余的时间。

相关参考:

- 第 512 页的『ts.ts_op_status – 表空间操作状态』
- 第 509 页的『ts.ts_util_auto_resize – 自动调整大小表空间利用率运行状况指示器』
- 第 477 页的『运行状况指示器』
- 第 507 页的『DMS 表空间的运行状况指示器』
- 第 512 页的『tsc.tscont_op_status – 表空间容器操作状态』
- 第 511 页的『tsc.tscont_util – 表空间容器利用率』
- 第 509 页的『ts.ts_auto_resize_status – 表空间自动调整大小状态运行状况指示器』

tsc.tscont_util – 表空间容器利用率

标识	tsc.tscont_util
运行状况监视器级别	表空间容器
类别	表空间存储器
类型	基于阈值上限
单位	百分比

描述 此运行状况指示器跟踪未使用自动存储器的每个 SMS 表空间的存储器消耗情况。如果对其定义容器的任何文件系统上都没有更多空间，则认为 SMS 表空间已满。

如果文件系统上没有可用空间可供扩展 SMS 容器，则表示关联表空间已满。

可对在用完可用空间的文件系统上定义的每个容器发出警报。

使用以下公式计算指示器:

```
(fs.used / fs.total)*100
```

其中 `fs` 是容器所在的文件系统。

SMS 表空间利用率以消耗空间的百分比来度量，其中高百分比指示未达到此指示器的最优运行状况。

此指示符的附加信息中包括的短期增长率和长期增长率可用来确定，当前增长率是短期畸变还是与长期增长一致。

附加信息中对离空间已满所余时间的计算可以预测到消耗所有可用空间所余的时间。

相关参考:

- 第 509 页的『ts.ts_util_auto_resize – 自动调整大小表空间利用率运行状况指示器』
- 第 477 页的『运行状况指示器』
- 第 512 页的『ts.ts_op_status – 表空间操作状态』
- 第 510 页的『ts.ts_util – 表空间利用率』
- 第 512 页的『tsc.tscont_op_status – 表空间容器操作状态』
- 第 509 页的『ts.ts_auto_resize_status – 表空间自动调整大小状态运行状况指示器』

ts.ts_op_status – 表空间操作状态

标识 ts.ts_op_status

运行状况监视器级别 表空间

类别 表空间存储器

类型 基于状态

单位 不适用

描述 表空间的状态可以限制可执行的活动或任务。从正常状态切换至另一状态可能生成“注意”警报。

相关参考:

- 第 477 页的『运行状况指示器』
- 『LIST TABLESPACES command』（*Command Reference*）
- 第 313 页的『tablespace_state – 表空间状态』
- 第 512 页的『tsc.tscont_op_status – 表空间容器操作状态』
- 第 511 页的『tsc.tscont_util – 表空间容器利用率』
- 第 510 页的『ts.ts_util – 表空间利用率』

tsc.tscont_op_status – 表空间容器操作状态

标识 tsc.tscont_op_status

运行状况监视器级别 表空间容器

类别 表空间存储器

类型 基于状态

单位 不适用

描述 此运行状况指示器跟踪表空间容器的可访问性。容器的可访问性可以限制可执行的活动或任务。如果容器不可访问，则可能生成“注意”警报。

相关参考:

- 第 328 页的『container_accessible – 容器的可访问性』
- 第 477 页的『运行状况指示器』
- 第 325 页的『tablespace_num_containers – 表空间中的容器数目』

- 第 511 页的『tsc.tscont_util – 表空间容器利用率』
- 第 512 页的『ts.ts_op_status – 表空间操作状态』
- 第 510 页的『ts.ts_util – 表空间利用率』

排序运行状况指示器

db2.sort_privmem_util – 专用排序内存利用率

标识	db2.sort_privmem_util
运行状况监视器级别	数据库
类别	排序
类型	基于阈值上限
单位	百分比

描述 如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，则认为排序状态良好。

此指示器跟踪专用排序内存的利用率。如果 db2.sort_heap_allocated（系统监视元素）>= *sheapthres*（DBM 配置参数），则排序可能未成为 *sortheap* 参数定义的已满排序堆，并且可能会生成警报。

使用以下公式计算指示器:

```
(db2.sort_heap_allocated / sheapthres)*100
```

“阈值后排序数”快照监视元素测量超过排序堆阈值后请求堆的排序数。此指示器的值显示在“其他详细信息”中，对此运行状况指示器指示问题的严重程度。

“使用的最大专用排序内存”快照监视元素保留实例的专用排序内存高水位标记。此指示器的值显示在“其他信息”中，指示自上次回收实例后在任一时间点使用的最大专用排序内存量。此值可用来帮助确定 *sheapthres* 的适当值。

相关参考:

- 第 514 页的『db.spilled_sorts – 溢出排序的百分比』
- 第 477 页的『运行状况指示器』
- 第 515 页的『db.max_sort_shrmem_util – 长期共享排序内存利用率』
- 第 513 页的『db.sort_shrmem_util – 共享排序内存利用率』

db.sort_shrmem_util – 共享排序内存利用率

标识	db.sort_shrmem_util
运行状况监视器级别	数据库
类别	排序
类型	基于阈值上限
单位	百分比

描述 如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，则认为排序状态良好。

此指示器跟踪共享排序内存的利用率。*sheapthres_shr* 数据库配置参数是硬限制。如果分配接近该限制，则会生成警报。

使用以下公式计算指示器：

```
(db.sort_shrheap_allocated / sheapthres_shr)*100
```

注意，如果 *sheapthres_shr* 设置为 0，则 *sheapthres* 充当共享排序堆阈值。

“使用的最大共享排序内存”快照监视元素保留数据库的共享排序内存高水位标记。此指示器的值显示在“其他信息”中，指示自数据库处于活动状态后在任一时间点使用的最大共享排序内存量。此值可用来帮助确定共享排序内存阈值的适当值。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配排序内存资源。如果对排序内存区启用自调整内存功能，则应配置此运行状况指示器以禁用阈值检查。

相关参考:

- 第 477 页的『运行状况指示器』
- 第 515 页的『db.max_sort_shrmem_util － 长期共享排序内存利用率』
- 第 514 页的『db.spilled_sorts － 溢出排序的百分比』
- 第 513 页的『db2.sort_privmem_util － 专用排序内存利用率』

db.spilled_sorts － 溢出排序的百分比

标识	db.spilled_sorts
运行状况监视器级别	数据库
类别	排序
类型	基于阈值上限
单位	百分比

描述 如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，则认为排序状态良好。

溢出至磁盘的排序可能导致严重的性能下降。如果发生这种情况，则会生成警报。

使用以下公式计算指示器：

$$(db.sort_overflows_{t} - db.sort_overflows_{t-1}) / (db.total_sorts_{t} - db.total_sorts_{t-1}) * 100$$

其中 *t* 是当前快照，而 *t-1* 是 1 小时以前的快照。系统监视元素 db.sort_overflows（基于 sort_overflows 监视元素）是用完排序堆并且可能需要磁盘空间以供临时存储器使用的总排序数。元素 db.total_sorts（基于 total_sorts 监视元素）是已执行的排序总数。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配排序内存资源。如果对排序内存区启用自调整内存功能，则应配置此运行状况指示器以禁用阈值检查。

相关概念:

- 『自调整内存』（*《性能指南》*）

相关参考:

- 第 515 页的『db.max_sort_shrmem_util － 长期共享排序内存利用率』
- 第 513 页的『db.sort_shrmem_util － 共享排序内存利用率』
- 第 513 页的『db2.sort_privmem_util － 专用排序内存利用率』
- 第 477 页的『运行状况指示器』
- 第 203 页的『sort_overflows － 排序溢出数』
- 第 201 页的『total_sorts － 总排序数』

db.max_sort_shrmem_util － 长期共享排序内存利用率

标识	db.max_sort_shrmem_util
运行状况监视器级别	数据库
类别	排序
类型	基于阈值下限
单位	百分比

描述 如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，则认为排序状态良好。

此指示器跟踪配置过度的共享排序堆，了解是否存在可以释放以用于 DB2 中的其他位置的资源。

使用百分比很低时，可能会生成警报。

使用以下公式计算指示器:

```
(db.max_shr_sort_mem / sheapthres_shr)*100
```

系统监视元素 db.max_shr_sort_mem（基于 sort_shrheap_top 监视元素）是共享排序内存使用情况的高水位标记。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配排序内存资源。如果对排序内存区启用自调整内存功能，则应配置此运行状况指示器以禁用阈值检查。

相关概念:

- 『自调整内存』（*《性能指南》*）

相关参考:

- 第 477 页的『运行状况指示器』
- 第 514 页的『db.spilled_sorts － 溢出排序的百分比』
- 第 513 页的『db2.sort_privmem_util － 专用排序内存利用率』
- 第 513 页的『db.sort_shrmem_util － 共享排序内存利用率』

数据库管理器（DBMS）运行状况指示器

db2.db2_op_status － 实例工作状态

标识	db2.db2_op_status
运行状况监视器级别	实例
类别	DBMS
类型	基于状态
单位	不适用

描述 如果实例状态未对正在执行的活动或任务产生限制，则认为该实例的运行状况正常。

状态可以是下列其中一项：“活动”、“停顿暂挂”、“已停顿”或“已关闭”。非“活动”状态可能会生成“注意”警报。

如果运行状况指示器进入“关闭”状态，运行状况监视器就无法执行 db2.db2_op_status 运行状况指示器的操作。例如，当指示器所监视的实例由于显式的停止请求或异常终止而进入不活动状态时，指示器就会进入“关闭”状态。如果要让实例在任何异常终止发生后自动重新启动，则可以配置故障监视器（**db2fm**）以保持该实例高度可用。

相关参考:

- 第 148 页的『db2_status － DB2 实例的状态』
- 第 477 页的『运行状况指示器』
- 第 516 页的『实例最高严重性警报状态』

实例最高严重性警报状态

标识	不适用。此运行状况指示器没有配置或建议支持。
运行状况监视器级别	实例
类别	DBMS
类型	基于状态
单位	不适用

描述 此指示器表示要监视的实例的累积警报状态。实例的警报状态是要监视的实例、实例数据库及数据库对象的最高警报状态。警报状态的顺序如下所示：

- 警报
- 警告
- 注意
- 正常

实例的警报状态确定 DB2 的整体运行状况。

相关参考:

- 第 477 页的『运行状况指示器』
- 第 516 页的『db2.db2_op_status － 实例工作状态』

数据库运行状况指示器

db.db_op_status – 数据库操作状态

标识	db.db_op_status
运行状况监视器级别	数据库
类别	数据库
类型	基于状态
单位	不适用

描述 数据库的状态可以限制可执行的活动或任务。状态可以是下列其中一项：活动、停顿暂挂、已停顿或已前滚。从活动切换至另一状态可能会生成“注意”警报。

相关参考:

- 第 152 页的『db_status – 数据库状态』
- 第 517 页的『数据库最高严重性警报状态』
- 第 477 页的『运行状况指示器』

数据库最高严重性警报状态

标识	不适用。此运行状况指示器没有配置或建议支持。
运行状况监视器级别	数据库
类别	数据库
类型	基于状态
单位	不适用

描述 此指示器表示要监视的数据库的累积警报状态。数据库的警报状态是数据库及其对象的最高警报状态。警报状态的顺序如下所示:

- 警报
- 警告
- 注意
- 正常

相关参考:

- 第 517 页的『db.db_op_status – 数据库操作状态』
- 第 477 页的『运行状况指示器』

维护运行状况指示器

db.tb_reorg_req – 需要重组

标识	db.tb_reorg_req
运行状况监视器级别	数据库

类别	数据库维护
类型	基于集合状态
单位	不适用

描述:

此运行状况指示器跟踪在数据库中重组表或索引的需要。表或对表定义的所有索引需要重组以消除碎片数据。重组将通过压缩信息并重构行或索引数据完成。此操作的结果是性能得到提高并且释放了表或索引中的空间。

通过在自动维护策略中指定要评估的表名，可以过滤此运行状况指示器评估的一组表。可通过使用“自动维护”向导来执行此操作。

可能会生成“注意”警报来指示需要重组。可通过将 AUTO_REORG 数据库配置参数设置为 ON 来自动进行重组。如果已经启用自动重组，则“注意”警报表示有一个或多个自动重组无法成功完成，或者有需要重组的表，但由于每个数据库分区的表大小超过可以进行脱机重组的最大表大小限定，使重组工作无法执行。有关需要注意的对象的列表，请参阅此运行状况指示器收集的详细信息。

相关参考:

- 『REORG INDEXES/TABLE command』（*Command Reference*）
- 第 477 页的『运行状况指示器』

db.tb_runstats_req – 需要收集统计信息

标识	db.tb_runstats_req
运行状况监视器级别	数据库
类别	数据库维护
类型	基于集合状态
单位	不适用

描述 此运行状况指示器跟踪收集数据库中的表及其索引的统计信息的需要。需要收集表和对表定义的所有索引的统计信息以缩短查询执行时间。

此运行状况指示器认为可使用 SQL 查询来限制这些表。附加信息中的作用域显示此查询用于系统表的子查询子句。

可能会生成“注意”警报来指示需要收集统计信息。可通过将 AUTO_RUNSTATS 数据库配置参数设置为 ON 以自动收集统计信息。如果启用了自动统计信息收集，则“注意”警报将指示未能成功完成一个或多个自动统计信息收集。

相关参考:

- 第 477 页的『运行状况指示器』
- 『RUNSTATS command』（*Command Reference*）

db.db_backup_req – 需要数据库备份

标识	db.db_backup_req

运行状况监视器级别 数据库

类别 数据库维护

类型 基于状态

单位 不适用

描述 此运行状况指示器跟踪在数据库中执行备份的需要。恢复策略中应包含定期备份以保护数据，从而避免在硬件或软件故障时丢失数据。

此运行状况指示器根据上次备份后的经历时间和更改的数据量来确定需要进行数据库备份的时间。

可能会生成“注意”警报来指示需要进行数据库备份。可通过将 AUTO_BACKUP 数据库配置参数设置为 ON 来自动进行数据库备份。如果启用了自动数据库备份，则“注意”警报将指示未能成功完成一个或多个自动数据库备份。

相关参考:

- 第 153 页的『last_backup － 上次备份时间戳记』

高可用性灾难恢复运行状况指示器

db.hadr_op_status － HADR 操作状态

标识 db.hadr_op_status

运行状况监视器级别 数据库

类别 高可用性灾难恢复

类型 基于状态

单位 不适用

描述 此运行状况指示器跟踪数据库的高可用性灾难恢复（HADR）操作状态。主服务器与备用服务器之间的状态可能为下列其中一项：已连接、拥塞或已断开连接。从已连接切换至另一状态可能会生成“注意”警报。

相关概念:

- 『高可用性灾难恢复概述』（*《数据恢复及高可用性指南与参考》*）

相关参考:

- 第 407 页的『hadr_connect_status － HADR 连接状态监视元素』
- 第 406 页的『hadr_state － HADR 状态监视元素』
- 第 519 页的『db.hadr_delay － HADR 日志延迟』
- 第 477 页的『运行状况指示器』

db.hadr_delay － HADR 日志延迟

标识 db.hadr_delay

运行状况监视器级别 数据库

类别 高可用性灾难恢复

类型　　基于阈值上限

单位　　分钟

描述　此运行状况指示器跟踪主数据库上的数据更改与备用数据库上的更改复制之间的当前平均延迟（以分钟计）。如果延迟值很大，则在主数据库发生故障后转移至备用数据库时可能导致数据丢失。如果延迟值很大，还可能意味着因为主数据库先于备用数据库而需要接管时所花的停机时间更长。

相关概念:

- 『高可用性灾难恢复概述』（*《数据恢复及高可用性指南与参考》*）

相关参考:

- 『hadr_syncmode － 处于对等状态时日志写操作的 HADR 同步方式配置参数』（*《性能指南》*）
- 第 519 页的『db.hadr_op_status － HADR 操作状态』
- 第 477 页的『运行状况指示器』

日志记录运行状况指示器

db.log_util － 日志利用率

标识	db.log_util
运行状况监视器级别	数据库
类别	日志记录
类型	基于阈值上限
单位	百分比

描述　此指示器跟踪在数据库中使用的总活动日志空间量。

日志利用率以消耗空间的百分比来度量，出现高百分比时可能会生成警报。

使用以下公式计算指示器:

```
(db.total_log_used / (db.total_log_used + db.total_log_available)) * 100
```

与日志有关的数据库配置参数的值显示在附加信息中，这些值显示日志的当前分配。附加信息还包括具有最旧活动事务的应用程序的标识。可强制此应用程序释放日志空间。

相关任务:

- 『用配置参数配置 DB2』（*《性能指南》*）

相关参考:

- 第 477 页的『运行状况指示器』
- 第 520 页的『db.log_fs_util － 日志文件系统利用率』

db.log_fs_util － 日志文件系统利用率

标识	db.log_fs_util
运行状况监视器级别	数据库

类别 日志记录

类型 基于阈值上限

单位 百分比

描述 日志文件系统利用率跟踪事务日志所在的文件系统的充满程度。如果文件系统上没有空间，则 DB2 可能无法创建新的日志文件。

日志利用率以消耗空间的百分比来度量。如果文件系统中的可用空间量降至最低（即高百分比利用率），则可能生成警报。

使用以下公式计算此指示器： *(fs.log_fs_used / fs.log_fs_total)*100*，其中 fs 是日志所在的文件系统。

与日志有关的数据库配置参数的值显示在附加信息中，这些值显示日志的当前分配。如果启用了 userexit，则还会显示附加详细信息。

如果显示在附加详细信息中的“在日志磁盘已满时停止”设置为“是”并且利用率达到 100%，则应尽快解决所有警报，以限制对不能落实事务的应用程序的影响，直到成功创建日志文件。

相关参考:

- 第 477 页的『运行状况指示器』
- 第 520 页的『db.log_util – 日志利用率』

应用程序并发性运行状况指示器

db.deadlock_rate – 死锁率

标识 db.deadlock_rate

运行状况监视器级别 数据库

类别 应用程序并行性

类型 基于阈值上限

单位 每小时产生的死锁数

描述 死锁率跟踪死锁出现在数据库上的比率以及应用程序遇到争用问题的等级。死锁可能是由下列情况导致的:

- 数据库发生锁定升级
- 在系统生成的行锁定已足够的情况下应用程序显式锁定了表
- 绑定时应用程序使用了不适当的隔离级别
- 目录表已被锁定以供可重复读
- 应用程序正以不同的顺序获取相同的锁定，从而导致死锁。

使用以下公式计算指示器:

$(db.deadlocks_{t} - db.deadlocks_{t-1})$

其中“t”是当前快照，而“t-1”是上一个快照，即距获取当前快照之前 60 分钟时获取的快照。

死锁率越高，可能生成警报的争用等级就越高。

相关参考:

- 第 477 页的『运行状况指示器』
- 第 522 页的『db.lock_escal_rate – 锁定升级率』
- 第 522 页的『db.locklist_util – 锁定列表利用率』
- 第 523 页的『db.apps_waiting_locks – 等待锁定的应用程序所占的百分比』

db.locklist_util – 锁定列表利用率

标识	db.locklist_util
运行状况监视器级别	数据库
类别	应用程序并行性
类型	基于阈值上限
单位	百分比

描述 此指示器跟踪要使用的锁定列表内存量。每个数据库有一个锁定列表，锁定列表包含由同时连接至数据库的所有应用程序挂起的锁定。这是对锁定列表内存设置的限制。一旦达到该限制，就会因为下列情况而使得性能下降:

- 锁定升级将行锁定转换为表锁定，从而降低了数据库中的共享对象的并行性。
- 因为应用程序等待有限数目的表锁定，所以应用程序间会出现更多死锁。因此将回滚事务。

当最大锁定请求数达到对数据库设置的限制时，将对应用程序返回错误。

使用以下公式计算指示器:

```
(db.lock_list_in_use / (locklist * 4096)) * 100
```

利用率以消耗内存的百分比来度量，出现高百分比表示状况不佳。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配锁定内存资源。如果对锁定内存区启用自调整内存功能，则应配置此运行状况指示器以禁用阈值检查。

相关概念:

- 『自调整内存』（*《性能指南》*）

相关参考:

- 第 521 页的『db.deadlock_rate – 死锁率』
- 第 477 页的『运行状况指示器』
- 第 522 页的『db.lock_escal_rate – 锁定升级率』
- 第 523 页的『db.apps_waiting_locks – 等待锁定的应用程序所占的百分比』

db.lock_escal_rate – 锁定升级率

标识	db.lock_escal_rate
运行状况监视器级别	数据库
类别	应用程序并行性

类型 基于阈值上限

单位 每小时产生的锁定升级数

描述 此指示器跟踪锁定已从行锁定升级至表锁定从而影响事务并行性的次数。

当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量，或者所有应用程序消耗的锁定列表空间达到总锁定列表空间时，锁定将会升级。可用锁定列表空间量由 *maxlocks* 和 *locklist* 数据库配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，应用程序将使用锁定列表中为其他应用程序分配的空间。每个数据库有一个锁定列表，锁定列表包含由同时连接至数据库的所有应用程序挂起的锁定。当整个锁定列表已满时，将发生错误。

使用以下公式计算指示器:

$(db.lock_escals_{t} - db.lock_escals_{t-1})$

其中“t”是当前快照，而“t-1”是上一个快照，即距获取当前快照之前 60 分钟时获取的快照。

死锁率越高，可能生成警报的争用等级就越高。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配锁定内存资源。如果对锁定内存区启用自调整内存功能，则应配置此运行状况指示器以禁用阈值检查。

相关概念:

- 『自调整内存』（《*性能指南*》）

相关参考:

- 第 521 页的『db.deadlock_rate – 死锁率』
- 第 477 页的『运行状况指示器』
- 第 522 页的『db.locklist_util – 锁定列表利用率』
- 第 523 页的『db.apps_waiting_locks – 等待锁定的应用程序所占的百分比』

db.apps_waiting_locks – 等待锁定的应用程序所占的百分比

标识 db.apps_waiting_locks

运行状况监视器级别 数据库

类别 应用程序并行性

类型 基于阈值上限

单位 百分比

描述 此指示器度量所有当前执行的等待锁定的应用程序所占的百分比。

高百分比可能指示应用程序遇到并行性问题，这对性能有负面影响。

使用以下公式计算指示器:

```
(db.locks_waiting / db.appls_cur_cons) *100)
```

相关参考:

- 第 521 页的『db.deadlock_rate – 死锁率』

- 第 477 页的『运行状况指示器』
- 第 522 页的『db.lock_escal_rate – 锁定升级率』
- 第 522 页的『db.locklist_util – 锁定列表利用率』

程序包和目录高速缓存，以及工作空间运行状况指示器

db.catcache_hitratio – 目录高速缓存命中率

标识 db.catcache_hitratio

运行状况监视器级别 数据库

类别 程序包和目录高速缓存，以及工作空间

类型 基于阈值下限

单位 百分比

描述 命中率是一个百分比，用于指示目录高速缓存对避免对磁盘上的目录的实际访问所起到的帮助作用。高命中率指示在避免实际磁盘 I/O 访问方面很成功。

使用以下公式计算指示器:

```
(1-(db.cat_cache_inserts/db.cat_cache_lookups))*100
```

相关参考:

- 第 477 页的『运行状况指示器』
- 第 524 页的『db.pkgcache_hitratio – 程序包高速缓存命中率』
- 第 525 页的『db.shrworkspace_hitratio – 共享工作空间命中率』

db.pkgcache_hitratio – 程序包高速缓存命中率

标识 db.pkgcache_hitratio

运行状况监视器级别 数据库

类别 程序包和目录高速缓存，以及工作空间

类型 基于阈值下限

单位 百分比

描述 命中率是一个百分比，用于指示程序包高速缓存对避免从系统目录重新装入静态 SQL 的程序包和段以及避免重新编译动态 SQL 语句所起到的帮助作用。高命中率指示在避免这些活动方面很成功。

使用以下公式计算指示器:

```
(1-(db.pkg_cache_inserts/db.pkg_cache_lookups))*100
```

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配程序包高速缓存内存资源。如果对程序包高速缓存内存区启用自调整内存功能，则应配置此运行状况指示器以禁用阈值检查。

相关概念:

- 『自调整内存』（*《性能指南》*）

相关参考:

- 第 524 页的『db.catcache_hitratio – 目录高速缓存命中率』
- 第 477 页的『运行状况指示器』
- 第 525 页的『db.shrworkspace_hitratio – 共享工作空间命中率』

db.shrworkspace_hitratio – 共享工作空间命中率

标识 db.shrworkspace_hitratio

运行状况监视器级别 数据库

类别 程序包和目录高速缓存，以及工作空间

类型 基于阈值下限

单位 百分比

描述 命中率是一个百分比，用于指示共享 SQL 工作空间对避免初始化要执行的 SQL 语句的各段所起到的帮助作用。高命中率指示在避免此操作方面很成功。

使用以下公式计算指示器:

```
(1-(db.shr_workspace_section_inserts/db.shr_workspace_section_lookups))*100
```

相关参考:

- 第 524 页的『db.catcache_hitratio – 目录高速缓存命中率』
- 第 477 页的『运行状况指示器』
- 第 524 页的『db.pkgcache_hitratio – 程序包高速缓存命中率』

内存运行状况指示器

db2.mon_heap_util – 监视器堆利用率

标识 db2.mon_heap_util

运行状况监视器级别 实例

类别 内存

类型 基于阈值上限

单位 百分比

描述 此指示器跟踪基于带有标识 SQLM_HEAP_MONITOR 的内存池的监视器堆内存的消耗。

对内存池标识 SQLM_HEAP_MONITOR 使用以下公式计算利用率:

```
(db2.pool_cur_size / db2.pool_max_size) * 100
```

。

一旦此百分比达到最大值 100%，监视器操作可能会失败。

相关参考:

- 第 526 页的『db.db_heap_util – 数据库堆利用率』
- 第 477 页的『运行状况指示器』

db.db_heap_util – 数据库堆利用率

标识	db.db_heap_util
运行状况监视器级别	数据库
类别	内存
类型	基于阈值上限
单位	百分比

描述 此指示器跟踪基于带有标识 SQLM_HEAP_DATABASE 的内存池的监视器堆内存的消耗。

对内存池标识 SQLM_HEAP_DATABASE 使用以下公式计算利用率:

```
(db.pool_cur_size / db.pool_max_size) * 100
```

。

一旦此百分比达到最大值 100%，查询和操作可能会因为没有堆可用而失败。

相关参考:

- 第 477 页的『运行状况指示器』
- 第 525 页的『db2.mon_heap_util – 监视器堆利用率』

联合运行状况指示器

db.fed_nicknames_op_status – 昵称状态

标识	db.fed_nicknames_op_status
运行状况监视器级别	数据库
类别	联合
类型	基于集合状态
单位	不适用

描述:

此运行状况指示器检查联合数据库中定义的所有昵称以确定是否存在无效昵称。如果数据源对象已删除或者已更改，又或者如果用户映射不正确，则昵称可能会无效。

如果在联合数据库中定义的任何昵称无效，则可能会生成“注意”警报。有关需要注意的对象的列表，请参阅此运行状况指示器收集的详细信息。

如果此运行状况指示器要检查昵称状态，则 FEDERATED 数据库管理器参数必须设置为 YES。

相关参考:

- 『ALTER NICKNAME statement』（*SQL Reference, Volume 2*）
- 『CREATE NICKNAME statement』（*SQL Reference, Volume 2*）
- 『CREATE USER MAPPING statement』（*SQL Reference, Volume 2*）

- 『DROP statement』（*SQL Reference, Volume 2*）

db.fed_servers_op_status － 数据源服务器状态

标识	db.fed_servers_op_status
运行状况监视器级别	数据库
类别	联合
类型	基于集合状态
单位	不适用

描述:

此运行状况指示器检查联合数据库中定义的所有数据源服务器以确定是否存在不可用的数据源服务器。如果数据源服务器停止、不再存在或者进行了错误的配置，则数据源服务器可能不可用。

如果在联合数据库中定义的任何昵称无效，则可能会生成“注意”警报。有关需要注意的对象的列表，请参阅此运行状况指示器收集的详细信息。

如果此运行状况指示器要检查数据源服务器状态，则 FEDERATED 数据库管理器参数必须设置为 YES。

相关参考:

- 『ALTER SERVER statement』（*SQL Reference, Volume 2*）
- 『ALTER USER MAPPING statement』（*SQL Reference, Volume 2*）
- 『CREATE USER MAPPING statement』（*SQL Reference, Volume 2*）

第 12 章 运行状况监视器接口

运行状况监视器接口

下表列示 API 的运行状况监视器接口:

表 833. 运行状况监视器: API

监视任务	API
捕获运行状况快照	db2GetSnapshot 获取快照类为 SQLM_CLASS_HEALTH 的快照
获取带有集合对象的完整列表的运行状况快照	db2GetSnapshot 获取快照类为 SQLM_CLASS_HEALTH 并且 agent_id 为 SQLM_HMON_OPT_COLL_FULL 的快照
捕获带有公式、附加信息和历史记录的运行状况快照	db2GetSnapshot 获取快照类为 SQLM_CLASS_HEALTH_WITH_DETAIL 的快照
捕获带有公式、附加信息、历史记录和集合对象的完整列表的运行状况快照	db2GetSnapshot 获取快照类为 SQLM_CLASS_HEALTH_WITH_DETAIL 并且 agent_id 为 SQLM_HMON_OPT_COLL_FULL 的快照
转换自描述数据流	db2ConvMonStream 转换监视器流
估计运行状况快照的大小	db2GetSnapshotSize 估计 db2GetSnapshot 输出缓冲区所需的大小

下表列示 CLP 命令的运行状况监视器接口:

表 834. 运行状况监视器接口: CLP 命令

监视任务	CLP 命令
捕获运行状况快照	GET HEALTH SNAPSHOT 命令
捕获带有公式、附加信息和历史记录的运行状况快照	GET HEALTH SNAPSHOT WITH DETAILS 命令

下表列示 SQL 函数的运行状况监视器接口:

表 835. 运行状况监视器接口: SQL 函数

监视任务	SQL 函数
数据库管理器级别运行状况信息快照	HEALTH_DBM_INFO
数据库管理器级别运行状况指示器快照	HEALTH_DBM_HI
数据库管理器级别运行状况指示器历史记录快照	HEALTH_DBM_HI_HIS
数据库级别运行状况信息快照	HEALTH_DB_INFO
数据库级别运行状况指示器快照	HEALTH_DB_HI
数据库级别运行状况指示器历史记录快照	HEALTH_DB_HI_HIS
数据库级别运行状况指示器集合快照	HEALTH_DB_HIC
数据库级别运行状况指示器集合历史记录快照	HEALTH_DB_HIC_HIS
表空间级别运行状况信息快照	HEALTH_TBS_INFO
表空间级别运行状况指示器快照	HEALTH_TBS_HI
表空间级别运行状况指示器历史记录快照	HEALTH_TBS_HI_HIS
表空间容器级别运行状况信息快照	HEALTH_CONT_INFO
表空间容器级别运行状况指示器快照	HEALTH_CONT_HI
表空间容器级别运行状况指示器历史记录快照	HEALTH_CONT_HI_HIS

相关参考:

- 『db2ConvMonStream API - Convert the monitor stream to the pre-version 6 format』 (*Administrative API Reference*)
- 『db2GetSnapshot API - Get a snapshot of the database manager operational status』 (*Administrative API Reference*)
- 『db2GetSnapshotSize API - Estimate the output buffer size required for the db2GetSnapshot API』 (*Administrative API Reference*)
- 『GET HEALTH SNAPSHOT command』 (*Command Reference*)

第 5 部分 附录

附录 A. DB2 数据库技术信息

DB2 技术信息概述

DB2 技术信息可通过下列工具和方法获得:

- DB2 信息中心
  - 主题
  - DB2 工具的帮助
  - 样本程序
  - 教程
- DB2 书籍
  - PDF 文件（可下载）
  - PDF 文件（在 DB2 PDF CD 中）
  - 印刷版书籍
- 命令行帮助
  - 命令帮助
  - 消息帮助
- 样本程序

IBM 定期提供文档更新。如果访问 ibm.com® 的 DB2 信息中心上的联机版本，则不必安装文档更新，这是因为 IBM 会维护此版本的更新。如果已经安装了 DB2 信息中心，则建议安装文档更新。IBM 提供新信息时，文档更新允许更新从 *DB2 信息中心 CD* 安装或从 Passport Advantage 下载的信息。

注: DB2 信息中心主题的更新频率比 PDF 书籍或硬拷贝书籍的更新频率高。要获取最新信息，安装可用的文档更新，或者参阅 ibm.com 上的 DB2 信息中心。

可以在线访问 ibm.com 上的其他 DB2 技术信息，如技术说明、白皮书和 Redbooks™。访问位于以下网址的 DB2 信息管理软件库站点: http://www.ibm.com/software/data/sw-library/。

文档反馈

我们非常重视您对 DB2 文档的反馈。如果您想就如何改善 DB2 文档提出建议，请将电子邮件发送至 db2docs@ca.ibm.com。DB2 文档小组会阅读您的所有反馈，但不能直接答复您。请尽可能提供具体的示例，这样我们才能更好地了解您所关心的问题。如果您要提供有关具体主题或帮助文件的反馈，请加上标题和 URL。

请不要用以上电子邮件地址与 DB2 客户支持机构联系。如果您遇到文档不能解决的 DB2 技术问题，请与您当地的 IBM 服务中心联系以获得帮助。

相关概念:

- 『DB2 信息中心的功能部件』（*DB2 在线信息中心*）
- 『Sample files』（*样本主题*）

相关任务:

- 『Invoking command help from the command line processor』(*Command Reference*)
- 『Invoking message help from the command line processor』(*Command Reference*)
- 第 538 页的『更新安装在计算机或内部网服务器上的 DB2 信息中心』

相关参考:

- 第 534 页的『PDF 格式的 DB2 技术资料库』

PDF 格式的 DB2 技术资料库

下列各表描述 IBM 出版物中心(网址为 网址为 www.ibm.com/shop/publications/order)提供的 DB2 库。

尽管这些表标识书籍有印刷版，但可能未在您所在国家或地区提供它们。

这些书籍中的信息对于所有 DB2 用户来说都是基础知识；不管您是程序员、数据库管理员还是使用 DB2 Connect 或其他 DB2 产品的人员，都将发现此信息很有用。

表 836. DB2 技术信息

书名	书号	是否提供印刷版
《管理指南：实施》	s151-0278	是
《管理指南：计划》	s151-0280	是
Administrative API Reference	SC10-4231	是
Administrative SQL Routines and Views	SC10-4293	否
Call Level Interface Guide and Reference, Volume 1	SC10-4224	是
Call Level Interface Guide and Reference, Volume 2	SC10-4225	是
Command Reference	SC10-4226	否
《数据移动实用程序指南和参考》	s151-0284	是
《数据恢复和高可用性指南与参考》	s151-0281	是
Developing ADO.NET and OLE DB Applications	SC10-4230	是
Developing Embedded SQL Applications	SC10-4232	是
Developing SQL and External Routines	SC10-4373	否
Developing Java Applications	SC10-4233	是
Developing Perl and PHP Applications	SC10-4234	否
Getting Started with Database Application Development	SC10-4252	是

表 836. DB2 技术信息 (续)

书名	书号	是否提供印刷版
《Linux 和 Windows 上的 DB2 安装和管理入门》	g151-0300	是
《消息参考：第 1 卷》	s151-0305	否
《消息参考：第 2 卷》	s151-0306	否
《迁移指南》	G151-0481	是
《Net Search Extender 管理和用户指南》 **注**：此文档的 HTML 不是从 HTML 文档 CD 安装的。	S151-0360	是
《性能指南》	s151-0279	是
Query Patroller Administration and User's Guide	GC10-4241	是
《DB2 客户机快速入门》	g151-0302	否
《DB2 服务器快速入门》	g151-0299	是
Spatial Extender and Geodetic Data Management Feature User's Guide and Reference	SC18-9749	是
SQL Guide	SC10-4248	是
SQL Reference, Volume 1	SC10-4249	是
SQL Reference, Volume 2	SC10-4250	是
《系统监视器指南和参考》	s151-0283	是
《故障诊断指南》	G151-0285	否
《Visual Explain 教程》	S151-0455	否
《新增内容》	s151-0307	是
XML Extender Administration and Programming	SC18-9750	是
《XML 指南》	s151-0282	是
XQuery Reference	SC18-9796	是

表 837. 特定于 DB2 Connect 的技术信息

书名	书号	是否提供印刷版
《DB2 Connect 用户指南》	s151-0304	是
《DB2 Connect 个人版快速入门》	G151-0431	是
《DB2 Connect 服务器快速入门》	g151-0303	是

表 838. WebSphere® 信息集成技术信息

书名	书号	是否提供印刷版
WebSphere Information Integration: Administration Guide for Federated Systems	SC19-1020	是

表 838. WebSphere® 信息集成技术信息 (续)

书名	书号	是否提供印刷版
WebSphere Information Integration: ASNCLP Program Reference for Replication and Event Publishing	SC19-1018	是
WebSphere Information Integration: Configuration Guide for Federated Data Sources	SC19-1034	否
WebSphere Information Integration: SQL Replication Guide and Reference	SC19-1030	是

注: DB2 发行说明提供特定于产品的发行版和修订包级别的附加信息。有关更多信息，请参阅相关链接。

相关概念:

- 第 533 页的『DB2 技术信息概述』

- 『关于发行说明』(*《发行说明》*)

相关任务:

- 第 536 页的『订购印刷版 DB2 书籍』

订购印刷版 DB2 书籍

如果您需要印刷版的 DB2 书籍，可以在许多(但不是所有)国家或地区在线购买。无论何时都可以从当地的 IBM 代表处订购印刷版的 DB2 书籍。注意，*DB2 PDF 文档* CD 上的某些软拷贝书籍没有印刷版。例如，*DB2 消息参考*的任何一卷都没有提供印刷版书籍。

只要支付一定费用，就可以从 IBM 获取 DB2 PDF 文档 CD，该 CD 包含许多 DB2 书籍的印刷版。根据您下订单的位置，您可能能够从 IBM 出版物中心在线订购书籍。如果在线订购在您所在国家或地区不可用，您总是可以从当地的 IBM 代表处订购印刷版 DB2 书籍。注意，并非 DB2 PDF 文档 CD 上的所有书籍都有印刷版。

注: 最新最完整的 DB2 文档保留在网址如下的 DB2 信息中心中: http://publib.boulder.ibm.com/infocenter/db2help/。

过程:

要订购印刷版的 DB2 书籍:

- 要了解您是否可从所在国家或地区在线订购印刷版的 DB2 书籍，可查看 IBM 出版物中心站点，网址为: http://www.ibm.com/shop/publications/order。必须先选择国家、地区或语言才能访问出版物订购信息，然后再按照针对您所在位置的订购指示信息进行订购。
- 要从当地的 IBM 代表处订购印刷版的 DB2 书籍:
  - 从下列其中一个 Web 站点找到当地代表处的联系信息:
    - IBM 全球联系人目录，网址为 www.ibm.com/planetwide

- IBM 出版物 Web 站点，网址为 http://www.ibm.com/shop/publications/order。必须先选择国家、地区或语言才能访问对应您的所在地的出版物主页。在此页面中访问“关于此站点”链接。
  – 请在致电时说明您想订购 DB2 出版物。
  – 请向您当地的代表提供想要订购的书籍的书名和书号。

相关概念:

• 第 533 页的『DB2 技术信息概述』

相关参考:

• 第 534 页的『PDF 格式的 DB2 技术资料库』

从命令行处理器显示 SQL 状态帮助

DB2 返回可作为 SQL 语句结果的条件的 SQLSTATE 值。SQLSTATE 帮助说明 SQL 状态和 SQL 状态类代码的含义。

过程:

要调用 SQL 状态帮助，打开命令行处理器并输入:

`? sqlstate` 或 `? class code`

其中，*sqlstate* 表示有效的 5 位 SQL 状态，*class code* 表示该 SQL 状态的前 2 位。

例如，`? 08003` 显示 08003 SQL 状态的帮助，而 `? 08` 显示 08 类代码的帮助。

相关任务:

- 『Invoking command help from the command line processor』（*Command Reference*）
- 『Invoking message help from the command line processor』（*Command Reference*）

访问不同版本的 DB2 信息中心

对于 DB2 版本 9 主题，DB2 信息中心 URL 为 http://publib.boulder.ibm.com/infocenter/db2luw/v9/。

对于 DB2 版本 8 主题，请访问以下版本 8 信息中心 URL: http://publib.boulder.ibm.com/infocenter/db2luw/v8/。

相关任务:

• 第 538 页的『更新安装在计算机或内部网服务器上的 DB2 信息中心』

以首选语言显示 DB2 信息中心中的主题

DB2 信息中心尝试以您在浏览器首选项中指定的语言显示主题。如果未提供主题的首选语言翻译版本，则 DB2 信息中心将显示该主题的英文版。

过程:

要在 Internet Explorer 浏览器中以您的首选语言显示主题:

1. 在 Internet Explorer 中，单击**工具** —> **Internet 选项** —> **语言...**按钮。“语言首选项”窗口打开。
2. 确保您的首选语言被指定为语言列表中的第一个条目。
   - 要将新语言添加至列表，单击**添加...** 按钮。

     注: 添加语言并不能保证计算机具有以首选语言显示主题所需的字体。
   - 要将语言移至列表顶部，选择该语言并单击**上移**按钮直到该语言成为语言列表中的第一个条目。
3. 清除浏览器高速缓存然后刷新页面以便以首选语言显示 DB2 信息中心。

要在 Firefox 或 Mozilla 浏览器中以首选语言显示主题:
1. 选择**工具** —> **选项** —> **语言**按钮。“语言”面板将显示在“首选项”窗口中。
2. 确保您的首选语言被指定为语言列表中的第一个条目。
   - 要将新语言添加至列表，单击**添加...** 按钮以从“添加语言”窗口中选择一种语言。
   - 要将语言移至列表顶部，选择该语言并单击**上移**按钮直到该语言成为语言列表中的第一个条目。
3. 清除浏览器高速缓存然后刷新页面以便以首选语言显示 DB2 信息中心。

在某些浏览器和操作系统组合上，可能还必须将操作系统的区域设置更改为您选择的语言环境和语言。

相关概念:
- 第 533 页的『DB2 技术信息概述』

更新安装在计算机或内部网服务器上的 DB2 信息中心

如果在本地安装了 DB2 信息中心，则可以下载已更新的主题。大多数主题底部的“最近一次更新日期”值指示该主题的当前级别。

要确定整个 DB2 信息中心是否存在更新，请查找“信息中心”主页上的“最近一次更新日期”值。将本地安装的主页中的值与 http://www.ibm.com/software/data/db2/udb/support/icupdate.html 上最新可下载更新的日期进行比较。如果提供了较新的可下载更新，就可以更新本地安装的信息中心。

更新在本地安装的 DB2 信息中心要求您:
1. 停止计算机上的 DB2 信息中心，然后以独立方式重新启动信息中心。如果以独立方式运行信息中心，则网络上的其他用户将无法访问信息中心，因而您可以下载和应用更新。
2. 使用更新功能来确定 IBM 是否提供了更新包。

   注: 在 CD 上也提供了更新。有关如何配置信息中心以从 CD 安装更新的详细信息，请参阅相关链接。

   如果提供了更新包，则使用更新功能来下载这些更新包。(更新功能只能用于独立方式。)
3. 停止独立信息中心，然后在计算机上重新启动 DB2 信息中心服务。

过程:

要更新安装在您的计算机或内部网服务器上的 DB2 信息中心:

1. 停止 DB2 信息中心服务。
   - 在 Windows 上，单击**开始** → **控制面板** → **管理工具** → **服务**。然后右键单击 **DB2 信息中心**服务，并选择**停止**。
   - 在 Linux 上，输入以下命令:
     ```
     /etc/init.d/db2icdv9 stop
     ```
2. 以独立方式启动信息中心。
   - 在 Windows 上:
     a. 打开命令窗口。
     b. 浏览至信息中心的安装路径。在缺省情况下，DB2 信息中心安装在 C:\Program Files\IBM\DB2 Information Center\Version 9 目录中。
     c. 使用 DB2 信息中心的标准路径运行 help_start.bat 文件:
        ```
        <DB2 Information Center dir>\doc\bin\help_start.bat
        ```
   - 在 Linux 上:
     a. 浏览至信息中心的安装路径。在缺省情况下，DB2 信息中心安装在 /opt/ibm/db2ic/V9 目录中。
     b. 使用 DB2 信息中心的标准路径运行 help_start 脚本:
        ```
        <DB2 Information Center dir>/doc/bin/help_start
        ```

   系统缺省 Web 浏览器将启动以显示独立信息中心。
3. 单击“更新”按钮（ ）。在信息中心的右边面板上，单击**查找更新**。将显示现有文档的更新列表。
4. 要启动下载进程，请检查您想要下载的选项，然后单击**安装更新**。
5. 在完成下载和安装进程后，单击**完成**。
6. 停止独立信息中心。
   - 在 Windows 上，使用 DB2 信息中心的标准路径运行 help_end.bat 文件:
     ```
     <DB2 Information Center dir>\doc\bin\help_end.bat
     ```

     注: help_end 批处理文件包含安全终止用 help_start 批处理文件启动的进程所需的命令。不要使用 `Ctrl-C` 或任何其他方法来终止 help_start.bat。
   - 在 Linux 上，使用 DB2 信息中心的标准路径运行 help_end 脚本:
     ```
     <DB2 Information Center dir>/doc/bin/help_end
     ```

     注: help_end 脚本包含安全终止用 help_start 脚本启动的进程所需的命令。不要使用任何其他方法来终止 help_start 脚本。
7. 重新启动 DB2 信息中心服务。
   - 在 Windows 上，单击**开始** → **控制面板** → **管理工具** → **服务**。然后右键单击 **DB2 信息中心**服务，并选择**启动**。
   - 在 Linux 上，输入以下命令:
     ```
     /etc/init.d/db2icdv9 start
     ```

更新后的 DB2 信息中心将显示新的主题和更新后的主题。

相关概念:

- 『DB2 信息中心安装选项』（《*DB2 服务器快速入门*》）

相关任务:

- 『使用 DB2 安装向导安装 DB2 信息中心（Linux）』（*《DB2 服务器快速入门》*）
- 『使用 DB2 安装向导安装 DB2 信息中心（Windows）』（*《DB2 服务器快速入门》*）

DB2 教程

DB2 教程帮助您了解 DB2 产品的各个方面。这些课程提供了逐步指示信息。

开始之前:

可从信息中心查看 XHTML 版的教程: http://publib.boulder.ibm.com/infocenter/db2help/ 。

某些课程使用了样本数据或代码。有关其特定任务的任何先决条件的描述，请参阅教程。

DB2 教程:

要查看教程，单击标题。

本机 XML 数据存储
: 设置 DB2 数据库以存储 XML 数据以及如何对本机 XML 数据存储执行基本操作。

《Visual Explain 教程》
: 使用 Visual Explain 来分析、优化和调整 SQL 语句以获取更好的性能。

相关概念:

- 『Visual Explain 概述』（*《管理指南: 实施》*）

DB2 故障诊断信息

有大量故障诊断和问题确定信息可帮助您使用 DB2 产品。

DB2 文档
: 故障诊断信息可在 DB2 信息中心的“DB2 故障诊断指南”或“支持和故障诊断”部分找到。可在该处找到有关如何使用 DB2 诊断工具和实用程序隔离和找出问题的信息、某些最常见问题的解决方案以及有关如何解决使用 DB2 产品时可能遇到的问题的建议。

DB2 技术支持 Web 站点
: 如果您遇到了问题并且想要获取查找可能的原因和解决方案的帮助，请参阅 DB2 技术支持 Web 站点。该“技术支持”站点具有指向最新 DB2 出版物、技术说明、授权程序分析报告（APAR 或错误修订）、修订包和其他资源的链接。可搜索此知识库并查找问题的可能解决方案。

  访问位于以下网址的 DB2 技术支持 Web 站点: http://www.ibm.com/software/data/db2/udb/support.html

相关概念:

- 『问题确定简介』（*《故障诊断指南》*）

- 第 533 页的『DB2 技术信息概述』

条款和条件

如果符合以下条款和条件，则授予您使用这些出版物的准用权。

个人使用： 只要保留所有的专有权声明，您就可以为个人、非商业使用复制这些出版物。未经 IBM 明确同意，您不可以分发、展示或制作这些出版物或其中任何部分的演绎作品。

商业使用： 只要保留所有的专有权声明，您就可以仅在企业内复制、分发和展示这些出版物。未经 IBM 明确同意，您不可以制作这些出版物的演绎作品，或者在您的企业外部复制、分发或展示这些出版物或其中的任何部分。

除非本准用权中有明确授权，不得把其他准用权、许可或权利（无论是明示的还是暗含的）授予其中包含的出版物或任何信息、数据、软件或其他知识产权。

当使用这些出版物损害了 IBM 的利益，或者根据 IBM 的规定，未正确遵守上述指导说明时，则 IBM 保留自主决定撤销本文授予的准用权的权利。

您不可以下载、出口或再出口本信息，除非完全遵守所有适用的法律和法规，包括所有美国出口法律和法规。

IBM 对这些出版物的内容不作任何保证。这些出版物“按现状”提供，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的关于适销和适用于某种特定用途的保证。

附录 B. 声明

IBM 可能不在所有国家或地区提供本文档中讨论的产品、服务或功能特性。有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可。您可以用书面方式将许可查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

有关双字节（DBCS）信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: International Business Machines Corporation“按现状”提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的有关非侵权、适销和适用于某种特定用途的保证。某些国家或地区在某些交易中不允许免除明示或暗含的保证。因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。IBM 可以随时对本资料中描述的产品和／或程序进行改进和／或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的：（i）允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及（ii）允许对已经交换的信息进行相互使用，请与下列地址联系:

IBM Canada Limited
Office of the Lab Director
8200 Warden Avenue
Markham, Ontario

L6G 1C7
CANADA

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。因此，在其他操作环境中获得的数据可能会有明显的不同。有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。此外，有些测量是通过推算而估计的，实际结果可能会有差异。本文档的用户应当验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示了目标和意愿而已。

本信息可能包含在日常业务操作中使用的数据和报告的示例。为了尽可能完整地说明这些示例，示例中可能会包括个人、公司、品牌和产品的名称。所有这些名称都是虚构的，与实际商业企业所用的名称和地址的任何雷同纯属巧合。

版权许可:

本信息可能包括源语言形式的样本应用程序，这些样本说明不同操作平台上的编程方法。如果是为按照在编写样本程序的操作平台上的应用程序编程接口（API）进行应用程序的开发、使用、经销或分发为目的，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。这些示例并未在所有条件下作全面测试。因此，IBM 不能担保或暗示这些程序的可靠性、可维护性或功能。

凡这些样本程序的每份拷贝或其任何部分或任何衍生产品，都必须包括如下版权声明:

©（*贵公司的名称*）（*年*）。此部分代码是根据 IBM 公司的样本程序衍生出来的。© Copyright IBM Corp. （*输入年份*）. All rights reserved.

商标

DB2 版本 9 文档库的各个文档中标识的公司、产品或服务名称可能是 International Business Machines Corporation 或其他公司的商标或服务标记。有关 IBM Corporation 在美国和／或其他国家的商标的信息在 http://www.ibm.com/legal/copytrade.shtml 中。

下列各项是其他公司的商标或注册商标，且已在 DB2 文档库中的至少一份文档中使用:

Microsoft®、Windows、Windows NT® 和 Windows 徽标是 Microsoft Corporation 在美国和／或其他国家或地区的商标。

Intel、Itanium®、Pentium® 和 Xeon® 是 Intel Corporation 在美国和／或其他国家或地区的商标。

Java™ 和所有基于 Java 的商标是 Sun Microsystems, Inc. 在美国和／或其他国家或地区的商标。

UNIX 是 The Open Group 在美国和其他国家或地区的注册商标。

Linux 是 Linus Torvalds 在美国和／或其他国家或地区的商标。

其他公司、产品或服务名称可能是其他公司的商标或服务标记。

索引

[A]

[B]

[C]

[D]

[F]

[G]

[H]

[J]

[K]

[L]

[M]

[N]

[P]

[Q]

[R]

[S]

[T]

[W]

[X]

[Y]

[Z]

A

B

C

D

E

F

G

H

I

L

M

N

T

U

V

W

X

与 IBM 联系

要与您所在国家或地区的 IBM 联系，请查看网址如下的 IBM 全球联系人目录: http://www.ibm.com/planetwide

要了解有关 DB2 产品的更多信息，请访问 http://www.ibm.com/software/data/db2/。

IBM

中国印刷

s151-0283-00



Spine information:

IBM DB2 DB2 版本 9 **系统监视器指南和参考** IBM